DYNAMICALLY COMPRESSED BAYESIAN HIDDEN MARKOV MODELS USING HAAR WAVELETS

By

JOHN WIEDENHOEFT

A dissertation submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Alexander Schliep

And approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October 2018

ABSTRACT OF THE DISSERTATION

Dynamically Compressed Bayesian Hidden Markov Models using Haar Wavelets

by JOHN WIEDENHOEFT

Dissertation Director:

Alexander Schliep

Hidden Markov models (HMM) have enjoyed a rich history of successes over the past decades. They have been applied to great effect in almost any conceivable segmentation task, from speech recognition and part-of-speech tagging, over financial time series analysis, to seismology and beyond. In bioinformatics, they are widely used for tasks such as gene finding, isochore classification and, most recently, detection of copy-number variation (CNV) in genomic data. Advances in biotechnology, such as high-resolution DNA microarrays and next-generation genome sequencing, have created data sets of millions and billions of values, presenting new challenges to the application of this classic. CNV detection from large genomic data sets is gaining momentum in research and diagnostics applications. As it often involves limited computational resources and time constraints, the importance of fast, accurate and low-memory approaches to HMM inference is obvious.

As statistical models, HMM depend on a large number of parameters, which have to be either provided *a priori* by the user or inferred from the data. Classic frequentist maximum likelihood (ML) techniques like Baum-Welch (Bilmes 1998; Rabiner 1989; Rabiner & Juang 1986) are not guaranteed to be globally optimal, i. e. they can converge to the wrong parameter values, which can limit the accuracy of the segmentation. Furthermore, the Viterbi algorithm (Viterbi 1967) only yields a single maximum a posteriori (MAP) segmentation given a parameter estimate (Forney 1973). Failure to consider the full set of possible parameters precludes alternative

interpretations of the data, and makes it very difficult to derive p-values or confidence intervals. Furthermore, these frequentist techniques have come under increased scrutiny in the scientific community.

Bayesian inference techniques for HMMs, in particular Forward-Backward Gibbs sampling (Chib 1996; Scott 2002), provide an alternative for CNV detection as well (Guha, Li & Neuberg 2006; Shah, Xuan, et al. 2006; Shah, Lam, et al. 2007). Most importantly, they yield a complete probability distribution of copy numbers for each observation. As they are sampling-based, they are computationally expensive, which is problematic especially for high-resolution data. Though they are guaranteed to converge to the correct values under very mild assumptions, they tend to do so slowly, which can lead to over-segmentation and mislabeling if the sampler is stopped prematurely. The difficulties encountered in Bayesian HMM inference are the reason most research has focused on the ML approach (Ghahramani 2001).

Another issue in practice is the need to specify hyperparameters for the prior distributions. Despite the theoretical advantage of making the inductive bias more explicit, this can be a major source of annoyance for the user. It is also hard to justify any choice of hyperparameters when insufficient domain knowledge is available.

Recent work by Mahmud & Schliep (2011) has focused on accelerating Forward-Backward Gibbs sampling through the introduction of compressed HMMs and approximate sampling. For the first time, Bayesian inference could be performed at running times on par with classic maximum likelihood approaches. It was based on a greedy spatial clustering heuristic, which yielded a static compression of the data into blocks, and block-wise sampling. Despite its success, several important issues remain to be addressed. The blocks are fixed throughout the sampling and impose a structure that turns out to be too rigid in the presence of variances differing between CN states. The clustering heuristic relies on empirically derived parameters not supported by a theoretical analysis, which can lead to suboptimal clustering or overfitting. Also, the method cannot easily be generalized for multivariate data. Lastly, the implementation was primarily aimed at comparative analysis between the frequentist and Bayesian approach, as opposed to overall speed.

To address these issues and make Bayesian CNV inference feasible even on a laptop, we propose the combination of HMMs with another popular signal processing technology: Haar

wavelets have previously been used in CNV detection (Ben-Yaacov & Eldar 2008), mostly as a preprocessing tool for statistical downstream applications (Wang & Wang 2007; Hsu et al. 2005; Nguyen et al. 2007, 2010; Huang et al. 2008) or simply as a visual aid in GUI applications (Wang, Meza-Zepeda, et al. 2004; Autio et al. 2003). A combination of smoothing and segmentation has been suggested as likely to improve results (Lai et al. 2005), and here we show that this is indeed the case. Wavelets provide a theoretical foundation for a better, dynamic compression scheme for faster convergence and accelerated Bayesian sampling. We improve simultaneously upon the typically conflicting goals of accuracy and speed, because the wavelets allow summary treatment of "easy" CN calls in segments and focus computational effort on the "difficult" CN calls, dynamically and adaptively. This is in contrast to other computationally efficient tools, which often simplify the statistical model or use heuristics. The required data structure can be efficiently computed, incurs minimal overhead, and has a straightforward generalization for multivariate data. We further show how the wavelet transform yields a natural way to set hyperparameters automatically, with little input from the user.

We implemented our method in a highly optimized end-user software, called HaMMLET. Aside from achieving an acceleration of several orders of magnitude, it exhibits significantly improved convergence behavior, has excellent precision and recall, and provides Bayesian inference within seconds even for large data sets. The accuracy and speed of HaMMLET also makes it an excellent choice for routine diagnostic use and large-scale re-analysis of legacy data.

In Chapter 1, we describe structural genomic variations, the main biological and medical application that motivates our work, as well as the experimental platforms, the type of data they create, the computational challenges and previous approaches to solve them. Chapters 2 and 3 review the main ingredients of our method, Hidden Markov Models and the Haar wavelet transform. Chapter 4 derives our method and investigates its properties, while presenting a proof-of-concept implementation as well as experimental evaluation. In Chapter 5, an improved implementation is presented to tackle genome-sized data, along with novel algorithms. In Chapter 6 we apply our method to real-world data. Chapter 7 presents an outlook on promising research directions.

Acknowledgements

I am deeply indebted to my PhD adviser, Alexander Schliep, for all the years of challenge, support, patience, encouragement, and overall fun. My sincere thanks also go to my lab mates Rajat Shuvro Roy, Md Pavel Mahmud, and Ivani de Oliveira Negrão Lopes, my collaborators Eric Brugel, Alex Cagan (Wellcome Trust Sanger Institute), Rimma Kozhemyakina and Rimma Gulevich (Siberian Branch of the Russian Academy of Sciences), Rodrigo Franco Toso (Microsoft), Scott Edwards and Allison Shultz (Harvard), as well as my previous mentors Roland Krause (University of Luxembourg), Oliver Eulenstein (Iowa State University) and Martin Vingron (Max Planck Institute for Molecular Genetics) for getting me on the way, and to Devdatt Dubhashi (Chalmers University of Technology) and Christian P. Robert (Université Paris-Dauphine) for insightful discussions. I am especially grateful to Janet Kelso, Svante Pääbo and everyone at the Max Planck Institute for Evolutionary Anthropology in Leipzig for their kind hospitality. I would also like to thank Eugene Fiorini for the fun of mentoring in the REU program and the annual Thanksgiving dinners. Last but not least, I am deeply grateful to my family, especially my wonderful wife Polina, for their unconditional love and support, for putting up with all this, and for making the Atlantic Ocean seem so much smaller. This thesis would not have been possible without you.

Since this work integrates passages from published and submitted material (Wiedenhoeft, Brugel & Schliep 2016a,b; Wiedenhoeft & Schliep 2017, 2018), which is collaborative in its nature, I decided to keep the plural form "we" throughout this thesis, in order to acknowledge the contributions of my co-authors. While individual contributions are always hard to disentangle, Chapter 6 specifically is joint work with Alex Cagan during a visit at MPI-EVA, with experiments performed by Rimma Gulevich and Rimma Kozhemyakina (Wiedenhoeft, Cagan, Kozhemjakina, et al. 2017; Wiedenhoeft, Cagan, Kozhemyakina, et al. 2018). For a full list of publications and submissions, see page 131.

Table of Contents

List of Illustrations

Chapter 1

Introduction

The main motivation behind the development of our method is the inference of genomic copy-number variation (CNV) from experimental data. In this chapter, we provide a short overview of the biological background, the role of CNV in human health, as well as the experimental and computational framework for their study.

1.1 Structural variation in genomes

As far as we can tell, all life on Earth, including humans, owes its origin, continued functioning and long-term survival to its genetic material, called the *genome* in its totality. This term refers to both its genetic information, as well as the physical medium of its storage.

On the physical level, the genome consists of four *nucleobases*, separated into two groups: two purine bases (adenine and guanine, A and G) and two pyrimidine bases (cytosine and thymine, C and T). These are linked together in a sequential manner to a backbone of *phosphate* groups and the 5-carbon sugar *deoxyribose* into a single strand of *deoxyribonucleic acid*, or *DNA* for short. Each unit of phosphate, deoxyribose and nucleobase is called a *nucleotide*. The propensity of each group of nucleobases to form hydrogen bonds between *complementary bases* (A with T, C with G) leads to a phenomenon called *hybridization*, in which two strands of complementary base sequences align together, winding around each other in a *double-stranded helix*. A set of such helices called *chromosomes* carries the genetic information in organisms.

On the information level, the genome uses the four-letter code A, C, G, and T (with U for

uracil replacing thymine in RNA molecules). Since the sugar and phosphate in each nucleotide is identical among nucleotides, the distinction between nucleotides and nucleobases is void from an information-theoretic standpoint, and both are called a *base* in general parlance. As DNA is typically double-stranded, a complementary *base-pair* is used as the basic unit of genetic information, similar to *bit* and *byte*. It is commonly abbreviated[1] as *b*, and used with SI prefixes. e.g. *kilobase (kb)* or *megabase (Mb)*. This provides a convenient way to express the size of any genomic segment.

The genetic information stored in a chromosome is very complex. The term *genes* typically refers to so-called protein-coding regions, i. e. sequences that are transcribed into mRNA, which serves as a template for protein synthesis, a process called gene expression. mRNA transcripts contain so-called untranslated regions on both ends (5'UTR and 3'UTR), which are involved in RNA splicing, transcription regulation and translation; though not part of the final protein, these sequences are genomically encoded and thus subject to changes in the chromosome as well. In more general terms, genes also include sequences that encode for ncRNAs (non-coding RNAs), such as transfer-RNA, ribosomal RNA, microRNA et cetera. Regulatory elements, such as promoters, operators, transcription factor binding sites (TFBS), enhancers and silencers, are binding sites for proteins which regulate whether and to what extend a gene is expressed at any given time. Elucidation of information content is still an ongoing endeavor, but the plethora of functions described here may serve as a glimpse into the intricacy of information encoded in this simple strand-like molecule.

The human genome is subject to a variety of physical processes which disrupt its architecture, commonly referred to as *structural variants (SV)*. Chromosomes can be merged or split, have their parts rearranged, duplicated, deleted, inverted or shuffled (Fig. 1.1); a large variety of types and mechanisms has been studied, see Hastings et al. (2009). Since genetic information is encoded in the DNA sequence, any structural variation changes its content. Among the many types of SV, *copy-number variants (CNV)* and *single-nucleotide polymorphisms (SNP)* account for the majority of observed cases, and are relevant for the remainder of this thesis; other SV such as inversions and translocations are neither of interest in this study nor can they be inferred by

[1] Conventions vary, as b is sometimes used instead of *nt* (for *nucleotide*) to denotes a single base, whereas *bp* denotes a basepair.

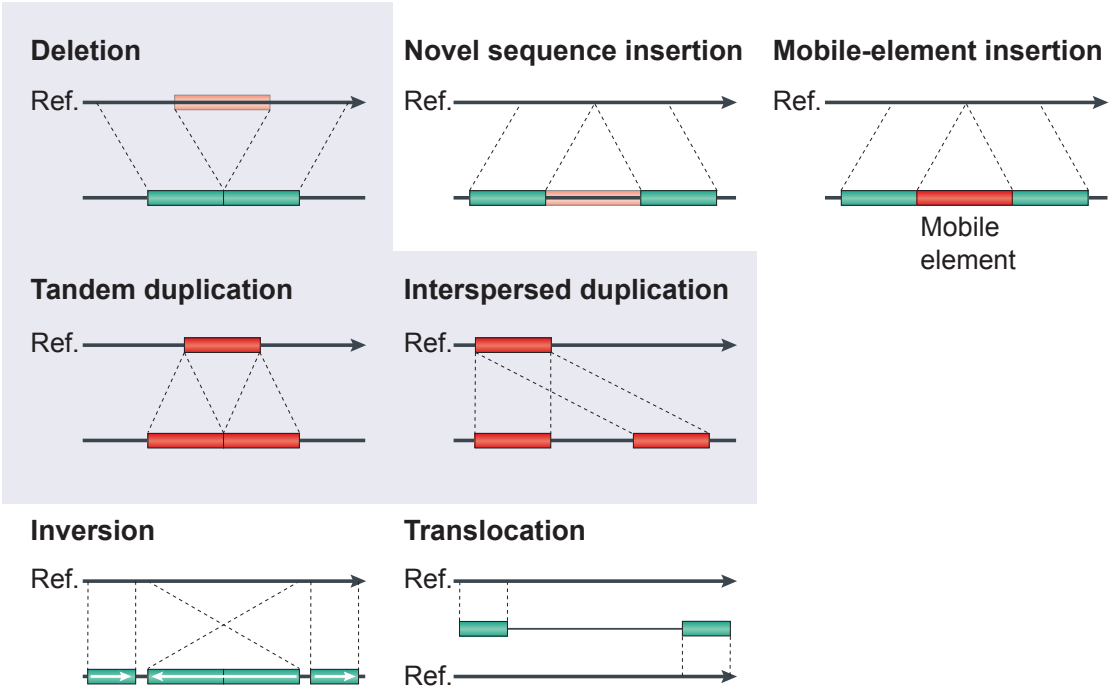


Figure 1.1: Overview of structural variations (SV) relative to a reference genome. Copy-number variants (CNV) detectable by the method presented in this thesis are indicated by gray background. Figure modified from Alkan, Coe & Eichler (2011), with permission from Macmillan Publishers Limited.

the algorithm presented here in its current form.

Copy-number variants (CNV) Copy number variants are defined as segments of DNA for which the number of occurrences within one individual's genome differs from that of corresponding sequences in the reference genome. In genomics, the term *copy-number variants* is commonly used to refer to variants found in multiple individuals of a population, and is distinguished from *copy-number aberrations*, which are typically associated with germline or somatic changes that have adverse effects for the health of one specific individual (Shlien & Malkin 2009; Valsesia et al. 2013; Navin 2015). Since CNV and CNA are equivalent from a data analysis standpoint, we refer to both of them as CNV in this thesis. Aside from large-scale changes due to full or partial *aneuploidy*, i. e. abnormal counts of entire or partial chromosomes, CNV are comprised of *insertions* and *deletions* (*indels*, for short), as well as *tandem repeats*. In the more narrow sense, CNV refers to intermediate-size copy-number aberrations between 1 kb and 5 Mb (Freeman 2006), which is small compared to the size of the data and thus challenging to find. Likewise,

indel often refers to much smaller events; this terminology historically stems from different experimental techniques and their resolution capacities. Unless otherwise stated, we shall refer to CNV as any kind of insertion or deletion of genomic material.

Single-nucleotide polymorphisms (SNP) Single-nucleotide polymorphisms refer to exchanges, losses or gains of single base pairs. They are important in the study of population genetics and human ancestry. While not directly a target of our computational method, they play an important role in the measurement of allelic CNV in the form of SNP arrays (Section 1.3.1).

Structural variations affect the information encoded on the DNA strands in a variety of ways; for a review, see Hurles, Dermitzakis & Tyler-Smith (2008). Insertions and deletions can disrupt a protein-coding sequence by introducing a premature stop codon, resulting in a truncated protein product. It can also create chimeric proteins, i. e. a molecule composed of two partial proteins. If the inserted sequence is not from a coding region, the resulting product would partially consist of a random amino acid sequence. Most of these changes will result in a loss of function for the affected gene, but especially chimeric proteins can result in coupling of regulatory pathways. Structural variants can also result in a translocation of regulatory elements such as promoters, which puts the affected gene under regulatory control of a pathway it would not usually be part of, thus disrupting the functioning of the cell on a systems level. CNV can also cause dosage effects, i. e. abnormal expression levels of gene products due to an abnormal number of genes. If SV occur in regions which do not code for genes, regulatory elements or other information such as ncRNA, and are not important for functions such as chromatin regulation, they may also have no effect at all.

The resulting changes to the genome can have phenotypic consequences, ranging from adaptation to environmental factors to severe illness. Studying SV promises to yield important insights into genome function and causal relationships between the genetic information level on one side and disease and phenotype on the other. The experimental platforms used in both clinical diagnostics as well as basic and translational research generate large amounts of high-resolution data, posing a challenge from the bioinformatics standpoint.

1.2 Implications of SV for human phenotype and health

Structural variations in genomes arise in two different forms. *Germinal* mutations occur in the germline, and usually affect all, or, in the case of mosaicism (Sturtevant 1929), some cells in an individual's body. *Somatic* mutations on the other hand arise within individual cells of an organism during its lifetime. These can be neutral to cell function, or, in most cases, incur damage that renders the cell inviable. In rare cases however, the changes can give the cell a competitive advantage by making it both self-sufficient and independent of internal and external regulatory and apoptosis-inducing signals, leading to uncontrolled cell proliferation known as *cancer*.

Structural variations have important biomedical implications, and are the focus of many genetic studies to elucidate the etiology of complex diseases, in particular cancer and neuropsychiatric disorders.

1.2.1 Somatic SV and cancer

It has been over a hundred years since an association between chromosome abnormalities and cancer has been established by Boveri (1914) as one of the most prominent features of cancer cells, encountered in all major types of tumors (Fröhling & Döhner 2008). This has lead to the notion of cancer being a disease of the genome. With cancer causing 1 in 8 deaths globally (Garcia et al. 2007), it is no exaggeration to say that elucidation of their role can be considered one of the most important research objectives of our time.

In cancer, somatic changes in copy number are one of the most noticeable changes in the affected cells (Nakagawa et al. 2015). Commonly referred to as copy-number aberrations (CNA), they are the target of diagnostic platforms such as array-based CGH (Section 1.3). While experimental evidence suggests a causative role for aneuploidy in tumorigenesis (Foijer, Draviam & Sorger 2008; Holland & Cleveland 2009), discerning causal CNV (driver mutations) from those that are mere byproducts of cancer progression (passenger mutations) is still an open research problem, which requires highly accurate calls of CNV from experimental platforms. For further details, Garraway & Lander (2013) has an excellent review of cancer genomics.

1.2.2 Germinal SV and human diversity

With the plethora of evidence for somatic variations, their germinal counterparts have only recently been determined to occur in abundance in the human population, constituting a major contributor to human genetic variation. While SNPs were originally thought to account for the majority of phenotypic variation (The International SNP Map Working Group et al. 2001; The International HapMap Consortium 2005), a number of recent studies have shown that CNV show remarkable abundance within the human population, even among healthy individuals (Sebat, Lakshmi, Troge, et al. 2004; Iafrate et al. 2004; Tuzun et al. 2005; Sharp et al. 2005; Fredman et al. 2004; Redon et al. 2006; de Vries et al. 2005; Schoumans et al. 2005; Sharp et al. 2005; Tyson et al. 2005; Conrad et al. 2006). Freeman (2006) contains an extensive history of CNV discovery. This discovery has huge implications outside the more obvious population genetics applications; the prevalence of CNV raises the issue of false negative calls in biomedical applications. Specifically, an accurate map of copy-number polymorphisms, stratified by donor ethnicity, is required as a true negative set when investigating disease associations of CNV.

1.2.3 Neuropsychiatric disorders

CNVs have been implicated in a variety of neuropsychiatric disorders (Malhotra & Sebat 2012; Cook Jr & Scherer 2008), in particular in autism (Chung, Tao & Tso 2014).

Autism spectrum disorder (ASD) refers to a range of neurodevelopmental disorder traditionally characterized by the triad of impaired reciprocal social behavior, poor communication skills, and stereotyped or repetitive patterns of behavior, interests or activities Kanner (1943). It includes previously distinguished diagnoses of Asperger syndrome, PDD-NOS, and childhood disintegrative disorder.

While ASD diagnosis is based entirely on behavioral observations due to the complex etiology of the phenomenon, it has long been recognized as having a strong genetic component (Folstein & Rosen-Sheidley 2001), but no single cause has yet been established. Early studies already showed microscopically visible chromosome anomalies in 7–8% of patients (Xu et al. 2004). With the advent of microarray technology (Section 1.3.1), a wide range of submicroscopic CNV

has been found to be associated with the ASD phenotype, and there is considerable support for a causative role of CNV. *De novo* occurrence of CNV has been reported as 3-5 times higher in affected families vs. control, and multiple de novo CNV have been found to lead to more severe symptoms. Furthermore, recurring CNV have been found in unrelated ASD patient. Lastly, CNV are enriched for genes associated with neuronal synaptic disorders, and overlap with those found in ADHD, SZ and ID; for an extensive review of those studies see Sener (2014). The role of CNVs in ASD and schizophrenia is reviewed in Merikangas, Corvin & Gallagher (2009).

However, there has been a paradigm shift in recent years towards the notion that there is no single causative mutation underlying the disease. Instead, the autism spectrum is considered to be a shared phenotype of multiple rare but highly penetrant genetic risk factors. To elucidate the role of CNV in this context, accurate CNV calls on high-resolution genomic data are critical.

Tourette syndrome (TS) is another disorder, characterized by motor tics, such as blinking, grimacing or head jerking, and phonic tics, such as coprolalia, palilalia and echolalia. It is often associated with obsessive-compulsive disorder (OCD) (Pauls et al. 1986) and attention-deficit hyperactivity disorder (ADHD). Though there is evidence for a genetic basis of the disease, it is not fully understood (O'Rourke et al. 2009). Recently, a role of rare large-scale CNV in TS and its aetiological overlap has been suggested (Nag et al. 2013), thereby adding to the notion of a general association of CNV and neuropsychiatric disorders.

1.3 Experimental platforms

A wide range of experimental platforms of increasing resolution has been developed over the last years (Alkan, Coe & Eichler 2011), and analysis of the data they produce is the central motivation and application area for our method in bioinformatics.

1.3.1 DNA microarrays

A DNA microarray, sometimes called biochip or gene chip, consists of a carrier substrate such as glass, plastic or silicone. On its surface, *probes* of single-strand DNA are affixed in a regular grid, such that each probe consists of DNA material of a different, predefined DNA sequence. Each probe is capable of capturing single-stranded DNA material of a reverse-complementary

sequence through *hybridization*, thus forming a short double-stranded helix. In order to assess the presence and abundance of DNA material in a sample, the sample DNA is fragmented into short pieces, fluorescently labeled and hybridized to the array. Then, the light emitted from each fluorescent spot is measured as a proxy of the abundance of DNA bound to each probe.

Probes are designed such that they represent known, preferably unique positions in the genome, so that the abundance of chromosomal segments can be measured. Microarrays yield very noisy data, due to the complexity of the biochemical process; furthermore, Gaussian noise is inevitable due to the quantum nature of photon emissions and its interaction with the light receptors in the equipment.

Obviously, the resolution of this method is limited by how many positions in the reference genome are covered by the set of probes. More importantly, microarrays rely on a predefined set of probes, i. e. they cannot detect sequences for which no probe is used. If, for instance, the genome contained a novel sequence due to a retrovirus, or the sample was contaminated by another DNA source, this would go undetected.

While there are many types of microarrays for different purposes, aCGH and SNP arrays are the most prominent types of platform for CNV detection.

Array-based Comparative Genomic Hybridization (aCGH) As the name suggests, aCGH is used to compare relative DNA abundance between two genomes, such as cancer tissue versus control. The two cases are labeled in different colors such as red and green, and then hybridized to the same microarray. The photo detector is then used to measure the resulting wavelength as a proxy for relative DNA abundance; for instance, if both genomes have the same amount at a probe, the resulting color will be yellow, whereas a shift towards red or green would indicate higher abundance in the respective genome. aCGH thus measures the relative abundance, though typically the control is assumed to be diploid, so that absolute abundance in the target genome can be derived. Relative abundance is typically expressed as the log2 ratio of green and red signals. The process is illustrated in Fig. 1.2.

It is important to note that since the chromosomes are fragmented, the only feasible way to position the measurements is via the reference genome used to design the probes. This means that the information obtained from aCGH is abundance of a chromosomal fragment, but not

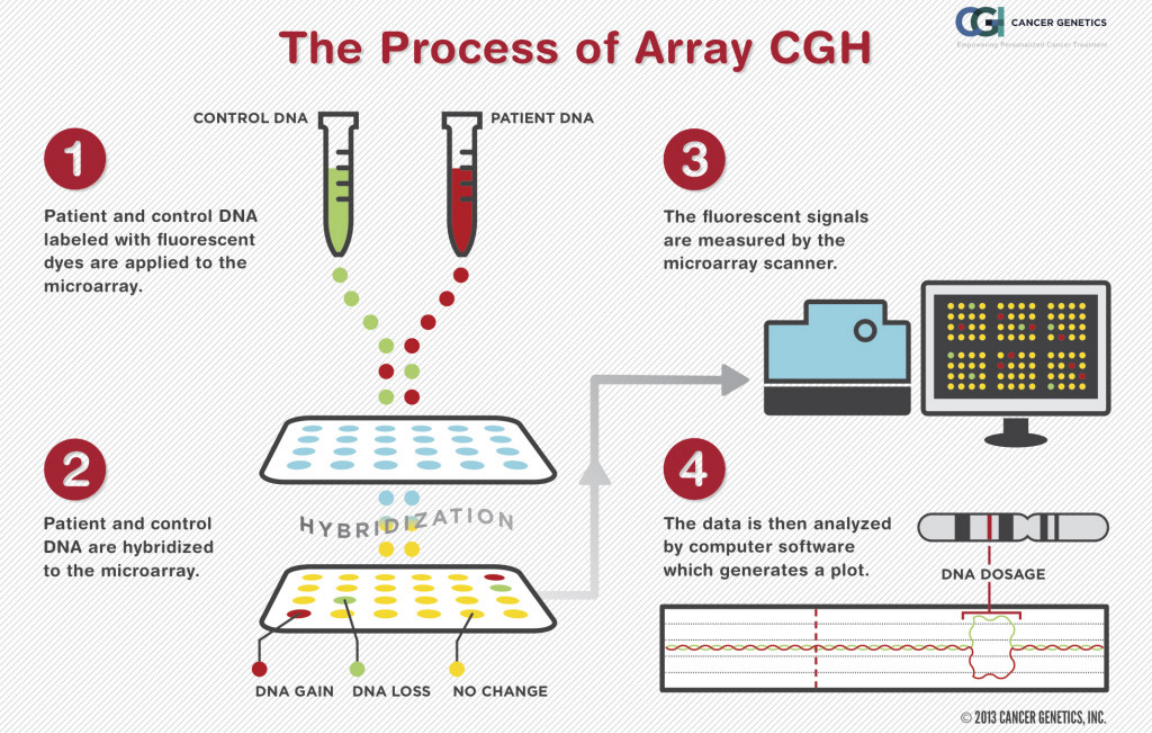


Figure 1.2: An overview of array-based comparative genome hybridization (aCGH). Sample and control DNA are flourescently labeled (1), and hybridized on an array of spotted DNA probes of known sequence, yielding a differential color signal (2). Measuring those colors and mapping the signal to the genome positions corresponding to each probe (3) yields a piecewise-constant signal with noise (4), which can be used to detect CNV. Figure reproduced with kind permission from Cancer Genetics, Inc. (www.cancergenetics.com).

its location. In the most extreme, hypothetical case that the entire genome had been severely shuffled and merged into a single chromosome, aCGH would not detect this.

Single-nucleotide polymorphism arrays (SNP arrays) The normal human genome contains two copies of each autosome. However, those two copies are usually not completely identical, but carry different alleles of the same locus, a phenomenon called *heterozygocity*. Most importantly, they often carry SNPs at many positions. However, some chromosomal segments may lack such differences; they are *homozygous*. Especially in somatic mutations, loss of heterozygocity (LOH), i. e. the loss of one allele coupled with a duplication of the other, is known to be associated with or even a driver of a wide range of diseases. Since loci which are heterozygous only in a SNP differ by only one nucleotide, and hence can bind both to the same probe. Therefore, LOH cannot be detected by aCGH. To alleviate this restriction, SNP arrays contain carefully designed sets of probes for SNPs known to exist in the human population, which allows to resolve

allelic copy number. The data from SNP arrays is similar to that from aCGH, with the important distinction that it is 2-dimensional, with each channel corresponding to one of two alleles. Since probes for each SNP differ only by one nucleotide, the resulting cross-hybridization yields a lower signal-to-noise ration than aCGH, thus posing additional challenges for CNV inference.

1.3.2 Next-generation sequencing

DNA sequencing refers to determining the sequence of bases A, C, G and T in a DNA sample. Aside from more labor-intensive methods such as Sanger sequencing and Maxam-Gilbert sequencing, *next-generation sequencing* (NGS) techniques have enabled high-throughput sequencing of DNA samples, creating several GB of data per sample; for a review, see Metzker (2010).

For NGS, DNA material is amplified and fragmented into short segments of several hundred bases. These fragments are then sequenced into short strings known as *reads*, ranging in size from 75-250 bp, depending on the platform. As the information about the original position of each read is lost during fragmentation, it has to be reconstructed computationally. In *paired-end sequencing*, the fragment is sequenced from both sides, yielding additional information about the relative distance of two reads. As this distance is limited by the fragment size, *mate-pair sequencing* uses an additional experimental step in which the two ends from a fragment originate from loci further apart in the genome than on the fragment.

When analyzing structural variation, it is important to note that reads essentially provide two layers of information. The pure sequence information can be used to determine sequence-level changes such as inversions and rearrangements, and can largely be subsumed under an elaborate approximate string matching paradigm, where the major source of noise are low-frequency sequencing errors. SV calls can often be made on a per-read basis, and are thus algorithmically more “localized”.

On the other hand, abundance of similar genomic sequences carries information about the number of copies of a genomic locus, which can be used for CNV detection. Two basic approaches exist: *Read-depth* (RD), sometimes referred to as *depth of coverage* (DOC) is determined by using a read mapper, essentially a highly optimized implementation of approximate string matching, to map reads to a known reference genome. After several post-processing steps to account for error sources such as amplification, GC and dinucleotide bias, the coverage at each nucleotide

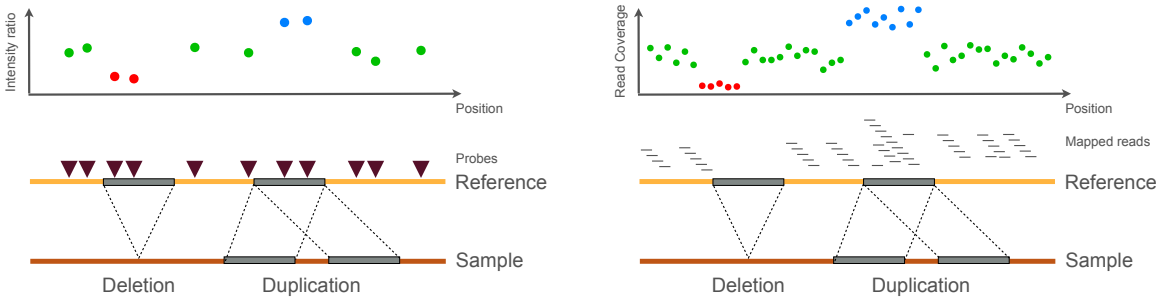


Figure 1.3: Illustration of aCGH (left) and read-depth data (right) for CNV detection. Duplication in the sample genome increases shifts the intensity ratio and the number of reads mapped to the affected loci in the reference genome, and hence induces an upwards shift in the count signal (blue). Likewise, a loss incurs a downwards shift in the signal (red).

in the reference is used as a proxy for DNA abundance at each position in the sample; for a review, see Magi et al. (2012). Read-depth data is illustrated in Fig. 1.3. *De novo assembly* (AS) on the other hand tries to merge the reads into contiguous strings called *contigs*, in which paired-end/mate-pair information as well as the multiplicity of occurrence of reads again used as a proxy for DNA abundance, but does not necessary require a reference genome. Both approaches require a more global look at the data, in that they should model the general noise level, for instance.

Whole-genome sequencing (WGS) The most straightforward application of NGS is the sequencing of the entire genomic material in a sample, known as *Whole-genome sequencing* (WGS). For reviews of this technique, see Pirooznia, Goes & Zandi (2015), Pabinger et al. (2014), and Hehir-Kwa, Pfundt & Veltman (2015). As it is not a targeted tool and yields information about every genomic locus regardless of the information it carries, it is the method of choice for fundamental and exploratory research, in particular for cancer (Nakagawa et al. 2015).

Whole-exome sequencing (WES) Due to the relatively high cost of WGS, targeted sequencing methods are often used, in which a predefined subset of DNA fragments is selected using target-enrichment techniques such as PCR, molecular inversion probes, in-solution capture, or hybridization to a microarray (hybrid capture). This reduces the amount of DNA to be sequenced significantly, especially in cases where only the protein-coding regions of the genome, the so-called *exome* is targeted. This *whole-exome sequencing* (WES), see Kadalayil et al. (2015), Tetreault et al. (2015), and Hehir-Kwa, Pfundt & Veltman (2015), leads to a 100-fold

reduction in sequencing for the human genome. While the reduced cost makes WES attractive for CNV detection, target enrichment introduces a considerable bias in read depth, which is not easily corrected computationally (Kadalayil et al. 2015). Though various approaches exist, we argue that the expected drop in sequencing cost makes it more worthwhile to focus method development on larger, but statistically more well-behaved data such as WGS.

1.4 Algorithmic challenges and prior approaches

Microarray-based methods will continue to play an important role in diagnostic settings due to their cost-effectiveness. The expected increase in resolution and low concordance among analytical tools (Pinto et al. 2011) necessitate the development of efficient bioinformatics tools for unbiased CNV calls on these platforms. However, the complexity of cancer and neuropsychiatric genomes necessitates even higher resolution that can only reasonably be achieved through sequencing approaches. In clinical settings, exome sequencing currently provides a compromise between cost and resolution, but the complex etiology of both classes of disease would require the use of WGS. It has been suggested that most driver genes, i. e. genes that are causal to tumorigenesis, have been identified (Nakagawa et al. 2015; Vogelstein et al. 2013; Lawrence et al. 2014), while rarer gene mutations and those in non-exomic regions such as promoters, ncRNAs and introns remain to be investigated (Leiserson et al. 2014; Garraway & Lander 2013). For instance, Freedman et al. (2011) suggests a role for regulatory elements in carcinogenesis. As a consequence, whole-genome approaches are likely to become more important in fundamental research (Garraway & Lander 2013), as the plethora of CNV calling methods for WGS data, reviewed in Pirooznia, Goes & Zandi (2015), Abel & Duncavage (2013), Zhao et al. (2013), Pabinger et al. (2014), and Duan et al. (2013), shows. While sequencing cost can be expected to drop, the computational power required to perform meaningful bioinformatics analysis is still beyond the capacities of most clinical institutions (Chung, Tao & Tso 2014), thus posing a huge challenge for the computer scientist.

While different platforms generate data of different characteristics, which in turn necessitate statistical and computational approaches tailored specifically to them (Wineinger et al. 2008), any experimental technology has its half-life, and generalized models that can treat data specifics

under a common framework are more likely to advance the field in the long run. As data created by microarrays and NGS for CNV detection can adequately be modeled as piecewise constant with additive noise (see Fig. 1.3), time-proven methods such as Hidden Markov Models (Baum & Petrie 1966) have repeatedly been applied in the context of CNV detection (Snijders, Fridlyand, et al. 2003; Sebat, Lakshmi, Troge, et al. 2004; Sebat, Lakshmi, Malhotra, et al. 2007; Fridlyand et al. 2004; Zhao 2004; de Vries et al. 2005; Nannya et al. 2005; Marioni, Thorne & Tavaré 2006; Korbel et al. 2007; Cahan et al. 2008; Rueda & Diaz-Uriarte 2009). They are favorable from a modeling standpoint, as they directly express the separate layers of observed measurements, such as log-ratios in array comparative genomic hybridization (aCGH), and their corresponding latent copy number (CN) states, as well as the underlying linear structure of segments. At the same time, advances in experimental technology create ever larger data sets, implying the need to leverage statistical analysis to handle big data. In this thesis, we shall hence take a high-level approach to CNV detection, and focus on improving and accelerating the type of Hidden Markov models suitable for this kind of data.

Chapter 2

Hidden Markov Models

In this chapter, we review the basics of HMM inference. We use the following notation: boldface ($\boldsymbol{y}$) denotes some ordered data type which allows indexing, such as a vector or an array. $\boldsymbol{y}[i]$ denotes the i-th element of $\boldsymbol{y}$; all indices are zero-based. $\boldsymbol{y}[i][j]$ denotes the element in the i-th row and j-th column in a two-dimensional data type. Slices/ranges are denoted as $\boldsymbol{y}[i:j]$ with the j-th entry included (closed interval). In contrast, $\boldsymbol{y}_i$ denotes the i-th element out of an (unordered) collection $\{\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots\}$.

2.1 Probability theory

Let the *probability space* $(\Omega, \Sigma, \mathbb{P})$ be a measure space, where Ω is called the *sample space*, Σ is the *set of events*, and $\mathbb{P}$ is called *probability measure*. Further, let $\mathbb{P}(\Omega) = 1$. A *random variable* is a function

$$X : (\Omega, \Sigma, \mathbb{P}) \to (\mathbb{K}, \mathcal{B}(\mathbb{K}), \mathbb{P}_X)$$

where $\mathbb{K}$ typically is $\mathbb{R}, \mathbb{Z}, \mathbb{C}, \ldots$. X is Σ-$\mathcal{B}(\mathbb{K})$-measurable, i. e. $\forall B \in \mathcal{B}(\mathbb{K}) : X^{-1}(B) \in \Sigma$, which implies

$$\mathbb{P}_X(B) = \mathbb{P}\left(X^{-1}(B)\right) = \mathbb{P}\left(\{\omega \in \Omega \,|\, X(\omega) \in B\}\right).$$

Further, let $\mathcal{P}_X(x)$ be a function

$$\mathcal{P}_X : \mathbb{R} \to [0, 1], \quad \int_{\mathbb{K}} \mathcal{P}_X(x)\, \mu(\mathrm{d}x) = 1,$$

called the *density of* X. The integral is the Lebesgue integral with an appropriate measure μ on $\mathbb{K}$. This function can be used to define the probability measure on $(\mathbb{K}, \mathcal{B}(\mathbb{K}), \mathbb{P}_X)$ as the integral

$$\mathbb{P}_X(A) := \int_A \mathcal{P}_X(x)\,\mu(\mathrm{d}x).$$

Since we will be dealing almost exclusively with simple densities and probability mass functions over subsets of $\mathbb{R}^n$ and $\mathbb{Z}^n$, we follow the notational convention common in machine learning to blur the distinction between events $(X = x)$, random variables (X), and their realizations/variates (x). We denote both the conditional density $\mathcal{P}(X \mid Y = y)$ as well as the likelihood function $\mathcal{P}(X = x \mid y)$ as $\mathcal{P}(X \mid Y)$, and it is derived from context which of X, Y remain fixed, whenever such information is necessary. Occasionally, the likelihood is denoted as $\mathcal{L}(Y \mid X)$. Consequently, there is no logical distinction between lower- and upper-case notation. Furthermore, the distinction between parameters and random variables is meaningless in a Bayesian context due to the *inversion principle*, see for example Robert (2007). We also follow the convention to denote the fact that a random variable X is distributed according to a distribution P depending on a parameter θ as $X \sim \mathcal{P}(X \mid \theta)$. We also simplify the integral notation to

$$\int_A \mathcal{P}_X(x)\,\mu(\mathrm{d}x) := \int \mathcal{P}(X)\,\mathrm{d}X\,.$$

In general, since the domain is often clear from context, we will drop it from the integral whenever possible, especially when integrating out dimensions, e.g.

$$\int \mathcal{P}(X, Y)\,\mathrm{d}Y = \mathcal{P}(X)\,.$$

Let

$$\mathbb{E}[X] := \int_\Omega X\,\mathrm{d}\mathbb{P} = \int_\Omega X(\omega)\mathbb{P}(\mathrm{d}\omega) = \int X\mathcal{P}(X)\,\mathrm{d}X$$

be the *expected value of random variable* X, or the *expectation* for short. For continuous and discrete univariate random variables, this becomes

$$\mathbb{E}[X] = \int_{\mathbb{R}} X\mathcal{P}(X)\,\lambda(\mathrm{d}X) = \int_{-\infty}^{\infty} X\mathcal{P}(X)\,\mathrm{d}X, \text{ and}$$
$$\mathbb{E}[X] = \int_{\mathbb{Z}} X\mathcal{P}(X)\,\#(\mathrm{d}X) = \sum_{X=-\infty}^{\infty} X\mathcal{P}(X),$$

with Lebesgue measure λ and counting measure $\#$, respectively. More generally, for any function f, we define the expectation

$$\mathbb{E}_X[f(X,Y)] \coloneqq \int f(X,Y)\mathcal{P}(X,Y)\,\mathrm{d}X,$$

as well as the *conditional expectation*

$$\mathbb{E}_X[f(X,Y)\,|\,Z] \coloneqq \int f(X,Y)\mathcal{P}(X,Y\,|\,Z)\,\mathrm{d}X.$$

By the *law of total expectation,*

$$\mathbb{E}[X] = \mathbb{E}_Y[\mathbb{E}_X[X\,|\,Y]].$$

Random variables X, Y are said to be *independent*, denoted $X \perp Y$, iff

$$\mathcal{P}(X,Y) = \mathcal{P}(X)\mathcal{P}(Y),$$

and conditionally independent, denoted $X \perp Y|Z$ iff

$$\mathcal{P}(X,Y\,|\,Z) = \mathcal{P}(X\,|\,Z)\mathcal{P}(Y\,|\,Z).$$

Linearity of expectation holds for any set of random variables X_i, i. e.

$$\mathbb{E}\left[\sum_i a_i X_i\right] = \sum_i a_i \mathbb{E}[X_i].$$

Similarly, the *variance* is defined as

$$\mathbb{V}[X] \coloneqq \mathbb{E}\left[(X - \mathbb{E}[X])^2\right] = \mathbb{E}\left[X^2\right] - \mathbb{E}[X]^2.$$

By Bienaymé's formula, for a sum of uncorrelated (and hence independent) random variables,

$$\mathbb{V}\left[\sum_i a_i X_i\right] = \sum_i a_i^2 \mathbb{V}[X_i].$$

A Bayes net (BN) is a directed acyclic graph which contains an edge from B to A if A is conditioned on B, see Fig. 2.2 for an example. For a random variable X, let

$$\begin{aligned}
\mathrm{pa}(X) &\coloneqq \{A\,|\,\mathcal{P}(X\,|A) \neq \mathcal{P}(X)\},\\
\mathrm{ch}(X) &\coloneqq \{A\,|\,\mathcal{P}(A|X) \neq \mathcal{P}(A)\}, \text{ and}\\
\mathrm{co}(X) &\coloneqq \{A\,|\,C \in \mathrm{ch}X, \mathcal{P}(C\,|A) \neq \mathcal{P}(C), A \neq X\}.
\end{aligned}$$

The quantities $\mathrm{pa}(X)$, $\mathrm{ch}(X)$ and $\mathrm{co}(X)$ are called the *parents*, *children*, and *coparents* of X respectively, referring to the relationships of notes in a Bayes net. Let

$$\partial X \coloneqq \mathrm{pa}(X) \cup \mathrm{co}(X) \cup \mathrm{ch}(X)$$

be the *Markov blanket of* X. Then

$$X \perp \partial X \mid (\{X\} \cup \partial X)^{\complement}.$$

The Markov blanket plays an important role in Gibbs sampling, since sampling of a random variable only requires conditioning on the Markov blanket as opposed to the entire set of variables in the model. An undirected graph is called a Markov Random Field (MRF) over a set of random variables if, for every variables, the set of its adjacent nodes equals the Markov blanket of said node.

2.2 Bayesian inference and the exponential family

Bayes' theorem can be easily derived in the following equivalent transformations

$$\begin{aligned}
\mathcal{P}(M, D) &= \mathcal{P}(D, M), \\
\mathcal{P}(M \mid D)\mathcal{P}(D) &= \mathcal{P}(D \mid M)\mathcal{P}(M), \text{ and} \\
\mathcal{P}(M \mid D) &= \frac{\mathcal{P}(D \mid M)\mathcal{P}(M)}{\mathcal{P}(D)}
\end{aligned}$$

If D is constant, then $\mathcal{P}(D)$ simply serves as a normalization factor for the left side to integrate to 1. Also, $\mathcal{P}(D \mid M)$ then becomes a function of M, not D, and does not represent a density since it does not necessarily integrate to 1. To emphasize this fact, $\mathcal{P}(D \mid M)$ is often denoted as $\mathcal{L}(M \mid D)$, yielding Bayes' theorem in the form of

$$\mathcal{P}(M \mid D) \propto_M \mathcal{L}(M \mid D)\mathcal{P}(M).$$

Here, $\mathcal{P}(M)$ is called the *prior distribution*, $\mathcal{L}(M \mid D)$ is called the *likelihood function*, and $\mathcal{P}(M \mid D)$ is called the *posterior distribution*; this captures the essence of Bayesian inference. If D represents some observed data, and M encodes our model of the data, the prior captures our model before the observation of the data, either from past observation or due to some assumptions (inductive

bias). The posterior represents the model after observing the data, and is proportional to the product of the prior and the probability of observing the data under the prior model. This process can be iterated, i. e. the posterior can become the prior for some new observations, hence the model gets more refined the more data is incorporated. In other words, the probability of the observed data under the current model becomes the likelihood of the model in light of the observed data.

Assume the model consists of a probability distribution fully described by some parameter θ, and the data consists of some variates $\boldsymbol{x}_i \sim \mathcal{P}(\boldsymbol{x} \mid \theta)$, so we obtain

$$\mathcal{P}(\theta \mid \boldsymbol{x}) \propto \mathcal{L}(\theta \mid \boldsymbol{x}) \mathcal{P}(\theta).$$

Further, assume the prior is fully defined by some *hyperparameter* $\boldsymbol{\tau}$. For the joint distribution of data, parameter and hyperparameter, we thus have

$$\begin{aligned} & \mathcal{P}(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{\tau}) = \mathcal{P}(\boldsymbol{x}, \boldsymbol{\theta}, \boldsymbol{\tau}) \\ \Leftrightarrow \qquad & \mathcal{P}(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{\tau}) \mathcal{P}(\boldsymbol{x}, \boldsymbol{\tau}) = \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{\theta}, \boldsymbol{\tau}) \mathcal{P}(\boldsymbol{\theta}, \boldsymbol{\tau}) \\ & \qquad\qquad = \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{\theta}, \boldsymbol{\tau}) \mathcal{P}(\boldsymbol{\theta} \mid \boldsymbol{\tau}) \mathcal{P}(\boldsymbol{\tau}). \end{aligned}$$

Assuming the hyperparameter does not influence the data directly, we have $\boldsymbol{x} \perp \boldsymbol{\tau} \mid \boldsymbol{\theta}$ and hence $\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{\theta}, \boldsymbol{\tau}) = \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{\theta})$, so we obtain

$$\mathcal{P}(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{\tau}) \propto_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{x}) \mathcal{P}(\boldsymbol{\theta} \mid \boldsymbol{\tau}) \mathcal{P}(\boldsymbol{\tau}).$$

The hyperparameter itself is kept constant, since it encodes our prior belief of the parameter model, hence

$$\mathcal{P}(\boldsymbol{\theta} \mid \boldsymbol{x}, \boldsymbol{\tau}) \propto_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{x}) \mathcal{P}(\boldsymbol{\theta} \mid \boldsymbol{\tau}).$$

Let an *exponential family distribution* (EFD) be any distribution for which the PDF is of the form

$$\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{\theta}) = \exp\left(\langle \boldsymbol{\theta}, \boldsymbol{T}(\boldsymbol{x}) \rangle + h(\boldsymbol{x}) - A(\boldsymbol{\theta})\right),$$

where $\boldsymbol{\theta}$ is a parameter vector[1], A is a scaling factor called the *log-partition function*, h is called the *carrier measure*, and $\boldsymbol{T}$ are the *sufficient statistics*. A vector $\boldsymbol{T}(\boldsymbol{X})$ is called a *sufficient statistic*

[1]The usual definition uses the more general form $\boldsymbol{\eta}(\boldsymbol{\theta})$ for the parameters, but here we always assume the canonical form $\boldsymbol{\eta}(\boldsymbol{\theta}) = \boldsymbol{\theta}$ for simplicity, since any EFD can be transformed accordingly.

of a random sample $\boldsymbol{X}$, if it contains all information about a distribution parameter that can be obtained from the complete sample. Formally,

$$\boldsymbol{\theta} \perp \boldsymbol{X} \mid \boldsymbol{T}(\boldsymbol{X}),$$

or, equivalently,

$$\mathcal{P}(\boldsymbol{\theta} \mid \boldsymbol{T}(\boldsymbol{X}), \boldsymbol{X}) = \mathcal{P}(\boldsymbol{\theta} \mid \boldsymbol{T}(\boldsymbol{X})).$$

By the Neyman-Fisher factorization theorem (Fisher 1922; Neyman 1936; Halmos & Savage 1949), a distribution has sufficient statistics if and only if its density function can be factored as

$$\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{\theta}) = H(\boldsymbol{x}) g(\boldsymbol{T}(\boldsymbol{x}, \boldsymbol{\theta}),$$

i. e. it depends on $\boldsymbol{x}$ only via $\boldsymbol{T}$. Obviously, EFD permit this factorization; furthermore, among all distributions for which the domain does not vary with the parameter, only EFD have sufficient statistics whose size remains bounded with increasing sample size (Darmois 1935; Koopman 1936; Pitman, Wishart & Fisher 1936).

For a sample of i.i.d. variates $\{\boldsymbol{x}_i\}_{i=1}^N$, the likelihood function of an EFD becomes

$$\mathcal{L}\left(\boldsymbol{\theta} \,\middle|\, \{\boldsymbol{x}_i\}_{i=1}^N\right) = \prod_{i=1}^N \mathcal{L}(\theta \mid \boldsymbol{x}_i) = \prod_{i=1}^N \exp\left(\langle \boldsymbol{T}(\boldsymbol{x}_i), \theta \rangle + h(\boldsymbol{x}_i) - A(\boldsymbol{\theta})\right) \tag{2.1}$$

$$= \exp\left(\left(\sum_{i=1}^N \langle \boldsymbol{T}(\boldsymbol{x}_i), \boldsymbol{\theta} \rangle + h(\boldsymbol{x}_i)\right) - NA(\boldsymbol{\theta})\right) \tag{2.2}$$

$$= \exp\left(\left\langle \sum_{i=1}^N \boldsymbol{T}(\boldsymbol{x}_i), \boldsymbol{\theta} \right\rangle + \sum_{i=1}^N h(\boldsymbol{x}_i) - NA(\boldsymbol{\theta})\right) \tag{2.3}$$

by linearity of the inner product in its first argument. It follows that the likelihood of a sample can be computed in a summary fashion, using only a fixed number of values in the sufficient statistics, independent of sample size N. For each EFD, there exists a *conjugate prior* distribution (Diaconis & Ylvisaker 1979) of the form

$$\mathcal{P}(\boldsymbol{\theta} \mid \tau, n) \propto \exp\left(\langle \tau, \boldsymbol{\theta} \rangle - nA(\boldsymbol{\theta})\right).$$

Dropping normalization constants for simplicity, the posterior can then be derived as

$$\mathcal{P}\left(\boldsymbol{\theta} \,\middle|\, \{\boldsymbol{x}_i\}_{i=1}^N, \tau\right) \propto \mathcal{L}\left(\boldsymbol{\theta} \,\middle|\, \{\boldsymbol{x}_i\}_{i=1}^N\right) \mathcal{P}(\boldsymbol{\theta} \mid \tau)$$

$$\propto \exp\left(\left\langle \sum_{i=1}^N \boldsymbol{T}(\boldsymbol{x}_i), \boldsymbol{\theta} \right\rangle - NA(\boldsymbol{\theta})\right) \exp\left(\langle \tau, \boldsymbol{\theta} \rangle - nA(\boldsymbol{\theta})\right)$$

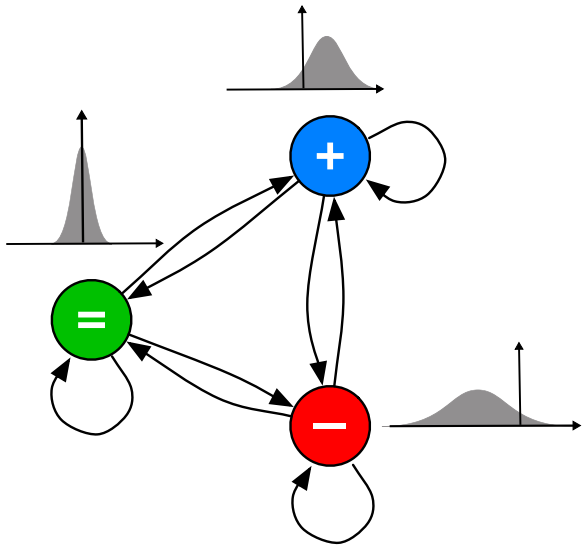


Figure 2.1: Visualization of a simple 3-state HMM for CNV detection. The hidden state path $\boldsymbol{q}$ is a Markov chain over the states $\{+,-,=\}$, representing a gain, loss, and diploid state, respectively. Upon visiting a state, a sample from its emission distribution (represented as gray densities next to each state) is drawn, according to the emission parameters $\boldsymbol{\theta} = \{\boldsymbol{\theta}_+, \boldsymbol{\theta}_-, \boldsymbol{\theta}_=\}$. In this case, emissions come from Gaussian distributions with different means and variances.

$$= \exp\left(\left\langle \sum_{i=1}^{N} \boldsymbol{T}(\boldsymbol{x}_i), \boldsymbol{\theta} \right\rangle + \langle \tau, \boldsymbol{\theta} \rangle - NA(\boldsymbol{\theta}) - nA(\boldsymbol{\theta})\right)$$

$$= \exp\left(\left\langle \tau + \sum_{i=1}^{N} \boldsymbol{T}(\boldsymbol{x}_i), \boldsymbol{\theta} \right\rangle - (n+N)A(\boldsymbol{\theta})\right).$$

Hence for posterior updates, it holds that

$$\mathcal{P}\left(\boldsymbol{\theta} \,\middle|\, \tau + \sum_{i=1}^{N} T(\boldsymbol{x}_i), n+N\right) \propto \mathcal{L}\left(\boldsymbol{\theta} \,\middle|\, \{\boldsymbol{x}_i\}_{i=1}^{N}\right) \mathcal{P}(\boldsymbol{\theta} \mid \tau, n).$$

Conjugacy thus implies that the posterior is of the same analytical form as the prior. Its *count parameter* n is updated by simply adding the sample size, and its *hyperparameter* τ is updated by adding the sufficient statistics of the observed sample. Since more observations can be included iteratively without changing the distribution type of the posterior, we say that a posterior is *strong* if its count parameter n is large.

2.3 The model itself

Let T be the length of the observation sequence. An HMM can be represented as a statistical model $(\boldsymbol{q}, \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi} \mid \boldsymbol{y})$, with transition matrix $\mathcal{A}$, a latent state sequence

$$\boldsymbol{q} = (\boldsymbol{q}[0], \boldsymbol{q}[1], \ldots, \boldsymbol{q}[T-1]),$$

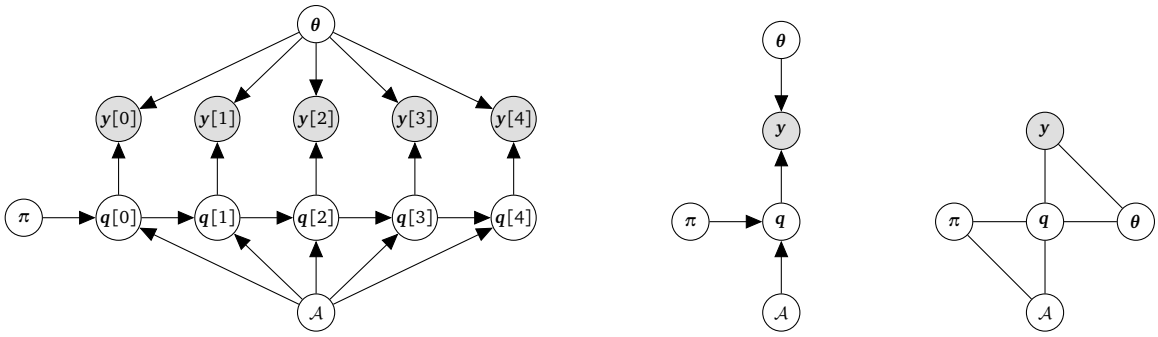


Figure 2.2: Left: Bayes net representation of an HMM. Arrows indicate conditional relations, e.g. $\mathcal{P}(\boldsymbol{q}[0]\,|\,\pi,\mathcal{A})$. Center: Bayes net representation of the HMM by treating all emissions $\boldsymbol{y}[t]$ and all latent state variables $\boldsymbol{q}[t]$ as joint probability vectors $\boldsymbol{y}$ and $\boldsymbol{q}$. Right: Markov Random Field (MRF) representation of the blocked HMM, obtained by adding the moralizing edges $(\boldsymbol{\theta},\boldsymbol{q})$ and $(\pi,\mathcal{A})$. The Markov blanket of a node corresponds to its direct neighbors in the MRF. Shading indicates that the variables are observed.

an observed emission sequence $\boldsymbol{y} = (\boldsymbol{y}[0], \boldsymbol{y}[1], \ldots, \boldsymbol{y}[T-1])$, emission parameters θ, and an initial state distribution $\boldsymbol{\pi}$. The vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_p)$ parametrizes the emission distributions depending on the underlying state, i. e.

$$\mathcal{P}(\boldsymbol{y}[t]\,|\,\boldsymbol{\theta},\boldsymbol{q}) = \mathcal{P}\big(\boldsymbol{y}[t]\,\big|\,\theta_{\boldsymbol{q}[t]}\big)\,.$$

The state sequence is modeled as a first-order Markov process with

$$\mathcal{P}(\boldsymbol{q}[t] = j\,|\,\boldsymbol{q}[t-1] = i) =: \mathcal{A}_{ij}$$

To derive conditional independence relations, the following Markov blankets can be easily read off (Fig. 2.2):

$$\partial\,\boldsymbol{y}[t] = \{\boldsymbol{q}[t], \boldsymbol{\theta}\}\,, \tag{2.4}$$

$$\partial\,\boldsymbol{q}[0] = \{\boldsymbol{\pi}, \boldsymbol{y}[0], \boldsymbol{q}[1], \mathcal{A}, \boldsymbol{\theta}\}\,, \tag{2.5}$$

$$\partial\,\boldsymbol{q}[t{>}0] = \{\boldsymbol{q}[t-1], \boldsymbol{q}[t+1], \boldsymbol{y}[t], \mathcal{A}, \boldsymbol{\theta}\}\,, \tag{2.6}$$

$$\partial\,\boldsymbol{\pi} = \{\boldsymbol{q}[0], \mathcal{A}\}\,, \tag{2.7}$$

$$\partial\,\mathcal{A} = \{\boldsymbol{q}, \boldsymbol{\pi}\}\,, \text{ and} \tag{2.8}$$

$$\partial\,\boldsymbol{\theta} = \{\boldsymbol{y}, \boldsymbol{q}\}\,. \tag{2.9}$$

Collapsing emissions and state sequence into one multivariate random variable each (Fig. 2.2) by merging nodes while maintaining adjacencies yields

$$\partial\,\boldsymbol{y} = \{\boldsymbol{q}, \boldsymbol{\theta}\}\,,$$

$$\partial\,\boldsymbol{\pi} = \{\boldsymbol{q}, \mathcal{A}\}\,, \text{ and}$$

$$\partial\,\boldsymbol{q} = \{\boldsymbol{y}, \boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{A}\}\,.$$

Note that this operation, while yielding a more convenient notation, induces unnecessary conditional relations, for instance within $\boldsymbol{q}$, which are understood to be ignored whenever appropriate.

Since we assume the hyperparameters to be constant, we drop dependence on them whenever appropriate for notational clarity, i. e. we define $\mathcal{P}(\boldsymbol{x}) \coloneqq \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{\tau}_{\boldsymbol{x}})$. The total distribution of an HMM then factorizes as

$$\mathcal{P}(\boldsymbol{y}, \boldsymbol{q}, \boldsymbol{\theta}, \boldsymbol{\pi}, \mathcal{A}) = \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{q})\,\mathcal{P}(\boldsymbol{q} \mid \boldsymbol{\pi}, \mathcal{A})\,\mathcal{P}(\boldsymbol{\theta})\,\mathcal{P}(\mathcal{A})\,\mathcal{P}(\boldsymbol{\pi})$$

$$= \mathcal{P}(\boldsymbol{\theta})\,\mathcal{P}(\mathcal{A})\,\mathcal{P}(\boldsymbol{\pi})\,\mathcal{P}(\boldsymbol{q}[0] \mid \boldsymbol{\pi}, \mathcal{A})\,\mathcal{P}(\boldsymbol{y}[0] \mid \boldsymbol{q}[0], \boldsymbol{\theta}) \prod_{t=1}^{T-1} \mathcal{P}(\boldsymbol{y}[t] \mid \boldsymbol{q}[t], \boldsymbol{\theta})\,\mathcal{P}(\boldsymbol{q}[t] \mid \boldsymbol{q}[t-1], \mathcal{A})\,.$$

From the MRF representation, it is easy to see

$$\mathcal{P}(\boldsymbol{y} \mid \boldsymbol{\theta}, \boldsymbol{q}, \mathcal{A}, \boldsymbol{\pi}) = \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{q}\,\boldsymbol{\theta})\,,$$

so the likelihood function for an HMM with observed data $\boldsymbol{y}$ is

$$\mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{y}) = \int \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{q}, \boldsymbol{\theta})\,\mathrm{d}\boldsymbol{q}\,.$$

2.4 Segmentation using HMM

Solving the segmentation task using HMM is based upon $\boldsymbol{q}$. There are two main approaches. For a fixed set of parameters, the most likely state sequence can be determined using the *Viterbi decoding algorithm*; such a state sequence is known as the *Viterbi path*. Alternatively, segmentation by *maximum posterior marginals (MPM)* chooses the most likely state at each position t. Notice that these two are *not* necessarily the same; indeed, a sequence of maximum state margins can have very low likelihood, or even be an impossible state sequence if it contains transitions for which $\mathcal{A}$ contains zero-entries. It does, however, have the advantage of integrating over all possible state paths.

The approaches central to our method are described in the following subsections. For notational clarity, all conditioning on HMM parameters is dropped wherever possible.

2.4.1 Filtering

Filtering describes the probability of being in a certain state at position t, given all observations up to this point, i. e.

$$\mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y}[0{:}t]).$$

Naïvely, for k states, that would require the computation of k^{t+1} state sequences, an unreasonably effort. However, a simple recursion based on the fact that state sequences can share a common prefix is used to derive a dynamic programming approach called *forward algorithm*. By Bayes' formula,

$$\mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y}[t], \boldsymbol{y}[0{:}t{-}1]) = \frac{\mathcal{P}(\boldsymbol{y}[t] \,|\, \boldsymbol{q}[t], \boldsymbol{y}[0{:}t{-}1])}{\mathcal{P}(\boldsymbol{y}[t] \,|\, \boldsymbol{y}[0{:}t{-}1])} \mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y}[0{:}t{-}1]).$$

Since

$$\boldsymbol{y}[t] \perp \boldsymbol{y}[0{:}t{-}1] \,|\, \boldsymbol{q}[t]$$

by the Markov property, we obtain

$$\boldsymbol{\alpha}_t[j] \coloneqq \mathcal{P}(\boldsymbol{q}[t] = j \,|\, \boldsymbol{y}[0{:}t]) \propto_j \mathcal{P}(\boldsymbol{y}[t] \,|\, \boldsymbol{q}[t] = j) \mathcal{P}(\boldsymbol{q}[t] = j \,|\, \boldsymbol{y}[0{:}t{-}1]).$$

This term is called the *forward variable*[2]. The distribution of the last term,

$$\mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y}[0{:}t{-}1]),$$

sometimes called the *one-step-ahead predictive density* (Murphy 2012), can easily be expressed as

$$\begin{aligned} \mathcal{P}(\boldsymbol{q}[t] = j \,|\, \boldsymbol{y}[0{:}t{-}1]) &= \sum_i \mathcal{P}(\boldsymbol{q}[t] = j \,|\, \boldsymbol{q}[t{-}1] = i) \mathcal{P}(\boldsymbol{q}[t{-}1] = i \,|\, \boldsymbol{y}[0{:}t{-}1]) \\ &= \sum_i \mathcal{A}_{ij} \boldsymbol{\alpha}_{t-1}[i]. \end{aligned}$$

The forward variables can thus be recursively defined as

$$\boldsymbol{\alpha}_t[j] \propto \mathcal{P}\big(\boldsymbol{y}[t] \,\big|\, \theta_j\big) \sum_i \mathcal{A}_{ij} \boldsymbol{\alpha}_{t-1}[i].$$

[2]Note that in other derivations, the forward variable refers to the joint density $\mathcal{P}(\boldsymbol{q}[t], \boldsymbol{y}[0{:}t])$ instead. In those settings, normalization to the conditional version used here is treated as a means to obtain numerical stability instead. The two versions are equivalent from a theoretical standpoint.

All forward variables for an HMM with k states can hence be recursively computed using a $T \times k$ dynamic programming matrix $\mathcal{T}$ called a *trellis*, using

$$\mathcal{T}[j][t] \propto \mathcal{P}\left(\boldsymbol{y}[t] \middle| \theta_j\right) \sum_i \mathcal{A}_{ij} \mathcal{T}[i][t-1]$$

with subsequent normalization of columns $\mathcal{T}[t]$ to sum to 1 in order to obtain the conditional state distribution. The forward recursion can be expressed compactly as

$$\boldsymbol{\alpha}_t \propto (\mathcal{A}^\intercal \boldsymbol{\alpha}_{t-1}) \odot \boldsymbol{\ell}_t,$$

where $\boldsymbol{\ell}_t$ is the vector of emission likelihoods at position t and $\odot$ is the Hadamard product. An implementation of this recursion has time complexity $O(k^2 T)$.

2.4.2 Smoothing

In order to obtain maximum state marginals, the distributions

$$\boldsymbol{\gamma}_t \coloneqq \mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y})$$

are derived from the forward variables

$$\mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y}[0{:}t]).$$

Using the fact that

$$\mathcal{P}(A \,|\, B, C) = \frac{\mathcal{P}(A, B \,|\, C)}{\mathcal{P}(B \,|\, C)} \propto_A \mathcal{P}(A, B \,|\, C),$$

the marginal state probability decomposes as

$$\begin{aligned} \mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y}) &\propto_{\boldsymbol{q}[t]} \mathcal{P}(\boldsymbol{y}[t{+}1{:}T{-}1], \boldsymbol{q}[t] \,|\, \boldsymbol{y}[0{:}t]) && (2.10) \\ &= \mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y}[0{:}t]) \mathcal{P}(\boldsymbol{y}[t{+}1{:}T{-}1] \,|\, \boldsymbol{q}[t], \boldsymbol{y}[0{:}t]) && (2.11) \\ &= \mathcal{P}(\boldsymbol{q}[t] \,|\, \boldsymbol{y}[0{:}t]) \mathcal{P}(\boldsymbol{y}[t{+}1{:}T{-}1] \,|\, \boldsymbol{q}[t]), && (2.12) \end{aligned}$$

since $\boldsymbol{y}[0{:}t] \perp \boldsymbol{y}[t{+}1{:}T{-}1] \,|\, \boldsymbol{q}[t])$. Let the *backward variable*

$$\boldsymbol{\beta}_t[i] \coloneqq \mathcal{P}(\boldsymbol{y}[t{+}1{:}T{-}1] \,|\, \boldsymbol{q}[t] = i),$$

then

$$\boldsymbol{\gamma}_t[i] \propto \boldsymbol{\alpha}_t[i] \boldsymbol{\beta}_t[i].$$

The backward variable can be recursively defined as

$$\boldsymbol{\beta}_{t-1} = \mathcal{P}(\boldsymbol{y}[t+1{:}T-1] \,|\, \boldsymbol{q}[t]) \tag{2.13}$$

$$= \sum_j \mathcal{P}(\boldsymbol{q}[t] = j, \boldsymbol{y}[t], \boldsymbol{y}[t+1{:}T-1] \,|\, \boldsymbol{q}[t-1]) \tag{2.14}$$

$$= \sum_j \mathcal{P}(\boldsymbol{y}[t+1{:}T-1] \,|\, \boldsymbol{q}[t] = j, \boldsymbol{y}[t], \boldsymbol{q}[t-1]) \mathcal{P}(\boldsymbol{q}[t] = j, \boldsymbol{y}[t] \,|\, \boldsymbol{q}[t-1]). \tag{2.15}$$

Since $\boldsymbol{y}[t+1{:}T-1] \perp (\boldsymbol{q}[t-1], \boldsymbol{y}[t]) \,|\, \boldsymbol{q}[t]$,

$$\boldsymbol{\beta}_{t-1} = \sum_j \mathcal{P}(\boldsymbol{y}[t+1{:}T-1] \,|\, \boldsymbol{q}[t] = i) \mathcal{P}(\boldsymbol{q}[t] = i, \boldsymbol{y}[t] \,|\, \boldsymbol{q}[t-1]) \tag{2.16}$$

$$= \sum_j \mathcal{P}(\boldsymbol{y}[t+1{:}T-1] \,|\, \boldsymbol{q}[t] = j) \mathcal{P}(\boldsymbol{y}[t] \,|\, \boldsymbol{q}[t] = j, \boldsymbol{q}[t-1]) \mathcal{P}(\boldsymbol{q}[t] = j \,|\, \boldsymbol{q}[t-1]).$$

(2.17)

Since $\boldsymbol{y}[t] \perp \boldsymbol{q}[t-1] \,|\, \boldsymbol{q}[t]$,

$$\boldsymbol{\beta}_{t-1}[i] = \sum_j \mathcal{P}(\boldsymbol{y}[t+1{:}T-1] \,|\, \boldsymbol{q}[t] = j) \mathcal{P}(\boldsymbol{y}[t] \,|\, \boldsymbol{q}[t] = j) \mathcal{P}(\boldsymbol{q}[t] = j \,|\, \boldsymbol{q}[t-1] = i) \tag{2.18}$$

$$= \sum_j \beta_t[j] \boldsymbol{\ell}_t[j] \mathcal{A}_{i,j} \tag{2.19}$$

The backward recursion can be expressed compactly as

$$\boldsymbol{\beta}_t = \mathcal{A}(\boldsymbol{\ell}_{t+1} \odot \boldsymbol{\beta}_{t+1}).$$

2.5 Frequentist inference and its caveats

Smoothing requires the values of the latent HMM parameters to be known; however, in most practical applications, only $\boldsymbol{y}$ is observed directly. In the usual frequentist approach, the state sequence $\boldsymbol{q}$ is inferred by first finding a maximum likelihood estimate of the parameters,

$$(\mathcal{A}_{\mathrm{ML}}, \theta_{\mathrm{ML}}, \boldsymbol{\pi}_{\mathrm{ML}}) = \arg \max_{(\mathcal{A}, \theta, \boldsymbol{\pi})} \mathcal{L}(\mathcal{A}, \theta, \boldsymbol{\pi} \,|\, \boldsymbol{y}),$$

using the Baum-Welsh algorithm (Bilmes 1998; Rabiner 1989). This is only guaranteed to yield local optima, as the likelihood function is not convex. Repeated random reinitialization are used to find "good" local optima, but there are no guarantees for this method. Then, the most likely state sequence given those parameters,

$$\widehat{\boldsymbol{q}} = \arg \max_{\boldsymbol{q}} \mathcal{P}(\boldsymbol{q} \,|\, \mathcal{A}_{\mathrm{ML}}, \theta_{\mathrm{ML}}, \boldsymbol{\pi}_{\mathrm{ML}}, \boldsymbol{y}),$$

is calculated using the Viterbi algorithm (Viterbi 1967). However, if there are only a few aberrations, i. e. there is imbalance between classes, the ML parameters tend to overfit the normal state which is likely to yield incorrect segmentation (Mahmud & Schliep 2011). Especially in CNV inference, where the rare, non-diploid states are more informative, this is problematic. Furthermore, alternative segmentations given those parameters are also ignored, as are the ones for alternative parameters.

2.6 Bayesian inference

The Bayesian approach does not assume $(\mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi})$ to be known. Instead, a joint prior distribution

$$\mathcal{P}(\boldsymbol{q}, \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi} \mid \tau)$$

parametrized by $\tau = (\tau_{\mathcal{A}}, \tau_{\boldsymbol{\theta}}, \tau_{\boldsymbol{\pi}})$ captures our subjective believe about the values of the latent variables $(\boldsymbol{q}, \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi})$; notice that the distribution of $\boldsymbol{q}$ is solely defined by $(\mathcal{A}, \boldsymbol{\pi}, \boldsymbol{\theta})$, see Fig. 2.2, and hence does not require a hyperparameter. Upon observing data $\boldsymbol{y}$, the joint posterior is

$$\mathcal{P}(\boldsymbol{q}, \mathcal{A}, \theta, \boldsymbol{\pi} \mid \boldsymbol{y}, \tau) \propto \mathcal{L}(\boldsymbol{q}, \mathcal{A}, \theta, \boldsymbol{\pi} \mid \boldsymbol{y}) \mathcal{P}(\boldsymbol{q}, \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi} \mid \tau).$$

The distribution of state sequences is computed directly by integrating out the emission and transition variables,

$$\mathcal{P}(\boldsymbol{q} \mid \boldsymbol{y}, \tau) = \int \int \int \mathcal{P}(\boldsymbol{q}, \mathcal{A}, \theta, \boldsymbol{\pi} \mid \boldsymbol{y}, \tau) \, \mathrm{d}\boldsymbol{\pi} \, \mathrm{d}\theta \, \mathrm{d}\mathcal{A}. \tag{2.20}$$

Since this integral is intractable, it has to be approximated using Markov Chain Monte Carlo techniques, i. e. drawing N samples,

$$(\boldsymbol{q}^{(i)}, \mathcal{A}^{(i)}, \theta^{(i)}, \boldsymbol{\pi}^{(i)}) \sim \mathcal{P}(\boldsymbol{q}, \mathcal{A}, \theta, \boldsymbol{\pi} \mid \boldsymbol{y}, \tau), \tag{2.21}$$

and subsequently approximating marginal state probabilities by their frequency in the sample

$$\mathcal{P}(\boldsymbol{q}[t] = s \mid \boldsymbol{y}, \tau) \approx \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(q^{(i)}[t] = s). \tag{2.22}$$

Thus, for each position t, we get a complete probability distribution over the possible states, solving the smoothing problem in a Bayesian setting. As before, this yields an MPM estimate

which can be used for segmentation. Notice that MCMC does *not* allow for the Viterbi path to be computed.

The method of choice used in this thesis combines *Gibbs sampling* of the HMM with *forward-backward sampling* of the state sequence, yielding an MCMC inference algorithm known as *Forward-Backward Gibbs sampling (FBG)*.

2.6.1 Gibbs sampling

As the marginals of each variable are explicitly defined by conditioning on the other variables, an HMM lends itself to Gibbs sampling, i. e. repeatedly sampling from the marginals $(\mathcal{A} \mid \boldsymbol{q}, \theta, \boldsymbol{y}, \boldsymbol{\pi})$, $(\theta \mid \boldsymbol{q}, \mathcal{A}, \boldsymbol{y}, \boldsymbol{\pi})$, $(\boldsymbol{\pi} \mid \mathcal{A}, \theta, \boldsymbol{y}, \boldsymbol{q})$, and $(\boldsymbol{q} \mid \mathcal{A}, \theta, \boldsymbol{y}, \boldsymbol{\pi})$, conditioned on the previously sampled values. Using Bayes' formula and conditional independence relations in Eq. (2.5)–Eq. (2.9), the sampling process can be written as

$$\mathcal{A} \sim \mathcal{P}(\mathcal{A} \mid \boldsymbol{\pi}, \boldsymbol{q}, \tau_{\mathcal{A}}) \quad \propto \mathcal{L}(\mathcal{A} \mid \boldsymbol{\pi}, \boldsymbol{q}) \mathcal{P}(\mathcal{A} \mid \tau_{\mathcal{A}}), \tag{2.23}$$

$$\theta \sim \mathcal{P}(\theta \mid \boldsymbol{q}, \boldsymbol{y}, \tau_{\theta}) \quad \propto \mathcal{L}(\theta \mid \boldsymbol{q}, \boldsymbol{y}) \mathcal{P}(\theta \mid \tau_{\theta}), \tag{2.24}$$

$$\boldsymbol{\pi} \sim \mathcal{P}(\boldsymbol{\pi} \mid \mathcal{A}, \boldsymbol{q}, \tau_{\pi}) \quad \propto \mathcal{L}(\boldsymbol{\pi} \mid \mathcal{A}, \boldsymbol{q}) \mathcal{P}(\boldsymbol{\pi} \mid \tau_{\pi}), \text{ and} \tag{2.25}$$

$$\boldsymbol{q} \sim \mathcal{P}(\boldsymbol{q} \mid \mathcal{A}, \boldsymbol{y}, \theta, \boldsymbol{\pi}), \tag{2.26}$$

where τ_x represents hyperparameters to the prior distribution $\mathcal{P}(x \mid \tau_x)$. Typically, each prior will be conjugate, which yields

$$\mathcal{A} \sim \mathcal{P}(\mathcal{A} \mid \tau_{\mathcal{A}}(\boldsymbol{\pi}, \boldsymbol{q})), \tag{2.27}$$

$$\theta \sim \mathcal{P}(\theta \mid \tau_{\theta}(\boldsymbol{q}, \boldsymbol{y})), \tag{2.28}$$

$$\boldsymbol{\pi} \sim \mathcal{P}(\boldsymbol{\pi} \mid \tau_{\pi}(\mathcal{A}, \boldsymbol{q})), \text{ and} \tag{2.29}$$

$$\boldsymbol{q} \sim \mathcal{P}(\boldsymbol{q} \mid \mathcal{A}, \boldsymbol{y}, \theta, \boldsymbol{\pi}). \tag{2.30}$$

Notice that the state sequence does not depend on any prior. The sampling of parameters is straightforward using their conjugate priors.

2.6.2 Forward-Backward sampling

There are several schemes available to sample $\boldsymbol{q} \mid \boldsymbol{y}$. A direct Gibbs sampling approach would use

$$\mathcal{P}(\boldsymbol{q}[t] \mid \boldsymbol{q}[\neg t], \boldsymbol{y}) = \mathcal{P}(\boldsymbol{q}[t] \mid \boldsymbol{q}[t-1], \boldsymbol{q}[t+1], \boldsymbol{y}[t])$$

to sample a state based on its neighbors ($[\neg t]$ means all positions *except* t). This, however, yields high autocorrelation and slow mixing properties; instead, Scott (2002) has argued strongly in favor of *forward-backward sampling* (Chib 1996), also known as *forward filtering, backward sampling* (Murphy 2012). Variations of this approach have been implemented for segmentation of aCGH data before (Mahmud & Schliep 2011; Shah, Xuan, et al. 2006). Consider the following chain rule factorization,

$$\begin{aligned}\mathcal{P}(\boldsymbol{q}\,|\,\boldsymbol{y}) &= \mathcal{P}(\boldsymbol{q}[T-1]\,|\,\boldsymbol{y})\prod_{t=1}^{T-1}\mathcal{P}(\boldsymbol{q}[t]\,|\,\boldsymbol{q}[t+1{:}T-1],\boldsymbol{y})\\ &= \mathcal{P}(\boldsymbol{q}[T-1]\,|\,\boldsymbol{y})\prod_{t=1}^{T-1}\mathcal{P}(\boldsymbol{q}[t]\,|\,\boldsymbol{q}[t+1],\boldsymbol{y})\\ &= \mathcal{P}(\boldsymbol{q}[T-1]\,|\,\boldsymbol{y})\prod_{t=1}^{T-1}\mathcal{P}(\boldsymbol{q}[t]\,|\,\boldsymbol{q}[t+1],\boldsymbol{y}[0{:}t]),\end{aligned}$$

where the first step follows from the Markov property of $\boldsymbol{q}$, and the second step follows from the conditional independence relation

$$\boldsymbol{q}[0{:}t] \perp \boldsymbol{y}[t+1{:}T-1]\,|\,\boldsymbol{q}[t+1].$$

By Bayes' formula,

$$\mathcal{P}(\boldsymbol{q}[t]\,|\,\boldsymbol{q}[t+1],\boldsymbol{y}[0{:}t]) \propto_{\boldsymbol{q}[t]} \mathcal{P}(\boldsymbol{q}[t+1]\,|\,\boldsymbol{q}[t],\boldsymbol{y}[0{:}t])\,\mathcal{P}(\boldsymbol{q}[t]\,|\,\boldsymbol{y}[0{:}t]),$$

and, since

$$\boldsymbol{q}[t+1] \perp \boldsymbol{y}[0{:}t]\,|\,\boldsymbol{q}[t],$$

this becomes

$$\mathcal{P}(\boldsymbol{q}[t]\,|\,\boldsymbol{q}[t+1],\boldsymbol{y}[0{:}t]) \propto_{\boldsymbol{q}[t]} \mathcal{P}(\boldsymbol{q}[t+1]\,|\,\boldsymbol{q}[t])\,\mathcal{P}(\boldsymbol{q}[t]\,|\,\boldsymbol{y}[0{:}t]).$$

Hence, we can recursively sample $\boldsymbol{q}\,|\,\boldsymbol{y}$ in a backward fashion, using a trellis of forward variables and the *backward-sampling recursion*

$$\mathcal{P}(\boldsymbol{q}[t]=i\,|\,\boldsymbol{q}[t+1]=j,\boldsymbol{y}[0{:}t]) \propto_i \mathcal{A}_{i,j}\boldsymbol{\alpha}_t[i],$$

with subsequent normalization. It can be computed within the trellis by updating the columns as

$$\mathcal{T}[i][t] \leftarrow \mathcal{T}[i][t]\mathcal{A}_{i,\boldsymbol{q}[t+1]},$$

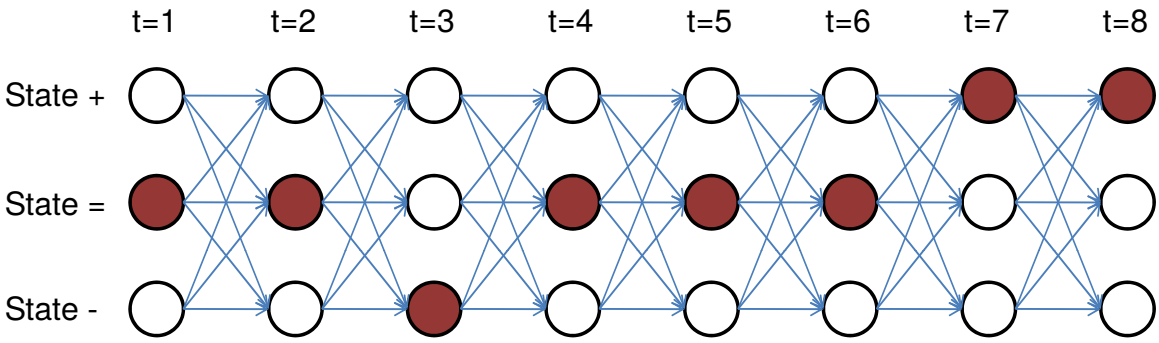


Figure 2.3: A small trellis for a simple HMM for CNV detection with three states $\{-, =, +\}$ and $T = 8$, yielding a 3×8 matrix. Entries are represented by circles. After an iteration of forward filtering, it contains the forward variables for each position and state. Relations used in the forward recursion are indicated by arrows; here, each forward variable depends on all previous forward variables through the transition matrix $\mathcal{A}$. After applying the backward algorithm, it contains the marginal state probabilities. If instead the backward sampling algorithm is used, it contains the sampling weight to obtain a state sequence variate $\boldsymbol{q}$, shown here by filled circles.

and sampling $\boldsymbol{q}[t]$ using the resulting weights, yielding a time complexity of $O(k^2T)$. The algorithm is illustrated in Fig. 2.3. The combination of this algorithm with the Gibbs sampling of the entire HMM is called *Forward-Backward Gibbs sampling* (FBG).

2.7 Compressed Hidden Markov Models

Forward-backward sampling of the state sequence, despite its various advantages over direct Gibbs (Scott 2002), is still expensive for genome-sized data. Firstly, notice that the forward variables need to be available for all states and data positions during Gibbs sampling; though this memoization in a dynamic programming table, called a *trellis* in the context of HMM, avoids exponential running times for the recursions shown above, it can grow to a considerable size. Secondly, since in each iteration a number of terms quadratic in the number of latent states has to be calculated at each position to obtain the forward variables, and a state has to be sampled at each position in the backward step, running times become impractical, especially due to billions of calls to a random number generator.

To alleviate these problems, Mahmud & Schliep (2011) have recently introduced *compressed FBG* for Gaussian emissions by sampling over a shorter sequence of sufficient statistics of data segments which are likely to come from the same underlying state, thereby decreasing the size of the trellis by a constant compression factor, at the cost of approximate sampling. Let

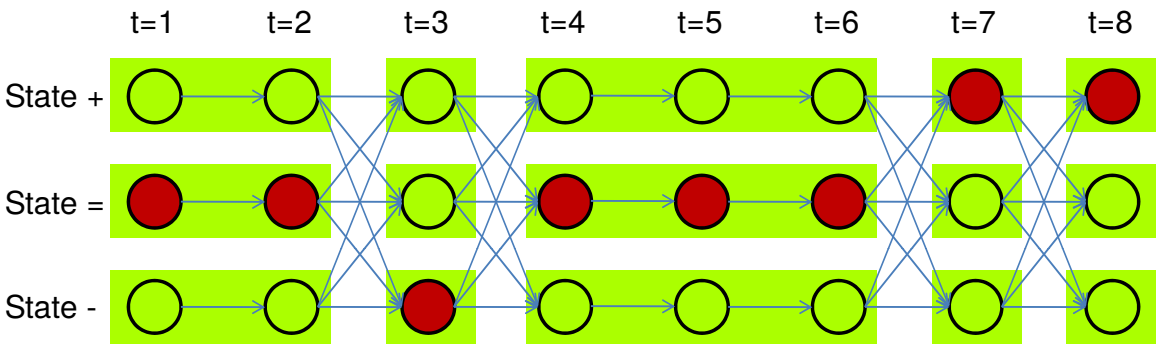


Figure 2.4: A compressed version of the trellis in Fig. 2.3 into 5 blocks. Notice that the number of recursive relations (arrows) is reduced, as all emissions within each block are assumed to have been generated by the same state. Notice that the sampled state sequence cannot change between states within a block.

$\boldsymbol{B} \coloneqq (B_w)_{w=1}^{W}$ be a partition of $\boldsymbol{y}$ into W blocks. Each block B_w contains n_w elements. Let $\boldsymbol{y}[w][k]$ the k-th element in B_w. The forward variable $\alpha_w(j)$ for this block needs to take into account the n_w emissions, the transitions into state j, and the $n_w - 1$ self-transitions, which yields

$$\alpha_w(j) \coloneqq A_{jj}^{n_w - 1} \mathcal{L}\left(\mu_j, \sigma_j^2 \,\middle|\, B_w\right) \sum_{i=1}^{n_w} \alpha_{w-1}(i) A_{ij}, \text{ and}$$

$$\mathcal{L}\left(\mu, \sigma^2 \,\middle|\, B_w\right) = \prod_{k=1}^{n_w} \mathcal{L}\left(\mu, \sigma^2 \,\middle|\, \boldsymbol{y}[w][k]\right).$$

This compression is illustrated in Fig. 2.4. The ideal block structure would correspond to the actual, unknown segmentation of the data. Any subdivision thereof would decrease the compression ratio, and thus the speedup, but still allow for recovery of the true breakpoints. In addition, such a segmentation would yield sufficient statistics for the likelihood computation that corresponds to the true parameters of the state generating a segment. Since the block structure is the target of the inference, the authors use a heuristic motivated by *kd*-trees to create a static block structure. The authors showed that the approximation error is small under reasonable state separation assumptions. In such a setting, the emitting state likelihood dominates the forward variables, whereas that of other states are close to zero, leading to negligible alternative state paths. This allows for an approximation of marginal probabilities, and MPM segmentation.

Chapter 3

Wavelet Transform

Our method uses the Haar wavelet transform to dynamically denoise and compress the input data, allowing the Gibbs sampler to operate on different resolution levels. In this chapter, we briefly review wavelet theory, closely following the exposition of Mallat (2009).

3.1 Multiresolution analysis and wavelets

Let (Ω, A, μ) be a measure space, where Ω is the base set, A is a σ-algebra over Ω, and μ is a measure on A. Let

$$L^p(\Omega, A, \mu) := \left\{ f : \Omega \to \mathbb{K}, \mathbb{K} \in \{\mathbb{R}, \mathbb{C}\}, f \text{ measurable}, \int_\Omega |f(x)|^p d\mu(x) < \infty \right\}$$

be the space of p-integrable functions. Let

$$L^2(\mathbb{R}) := L^2(\mathbb{R}, \mathcal{B}(\mathbb{R}), \lambda)$$

be the space of square-integrable functions over the reals with Borel algebra $\mathcal{B}(\mathbb{R})$ and the Lebesgue measure λ, and likewise for subintervals of $\mathbb{R}$. This is a Hilbert space with inner product

$$\langle f, g \rangle := \int_{\mathbb{R}} f(x) g(x) d\lambda(x) = \int_{-\infty}^{\infty} f(x) g(x) dx.$$

We are only concerned with functions over subsets of $\mathbb{R}$, so the inner product commutes without involving the complex conjugate. The inner product induces the norm

$$\|f\| := \sqrt{\langle f, f \rangle}.$$

Two functions f, g are said to be *orthogonal* iff $\langle f, g \rangle = 0$, and a function f is called *normal* iff $\|f\| = 1$. In signal processing terms, square-integrability

$$\int_{-\infty}^{\infty} |f(x)|^2 dx = \langle f, f \rangle < \infty$$

means that f has finite energy. $L^2(\mathbb{R})$ is a *separable* Hilbert space, i. e. it can be spanned by an orthonormal basis, a set of basis functions v_i and a set of coefficients a_i such that, for each $f \in L^2(\mathbb{R})$,

$$f(t) = \sum_{i=-\infty}^{+\infty} a_i v_i(t) \quad \text{with} \quad a_i = \langle f, v_i \rangle,$$
$$\forall i \neq j : \langle v_i, v_j \rangle = 0 \quad \text{(orthogonality), and}$$
$$\forall i : \|v_i\| = 1 \quad \text{(normality).}$$

The goal of *multiresolution analysis (MRA)* (Mallat 1989; Meyer & Salinger 1992) is to provide a sequence of well-behaved, nested subspaces, which allow to approximate a function $f \in L^2(\mathbb{R})$ at different levels of resolution. The differences between those resolution levels will be captured by *wavelet functions*. An example for the Haar wavelet is shown in Fig. 3.1. Consider an *approximation space* $\mathbb{V}_j \subset L^2(\mathbb{R})$, with *resolution* 2^{-j}, and its inverse 2^j, called *scale*. For any function $f \in L^2(\mathbb{R})$, its projection $P_{\mathbb{V}_j} f$ onto $\mathbb{V}_j$ provides a lower-resolution approximation. Let $\mathbb{V}_j$ be translation-invariant for integer multiples of scale 2^j, i. e.

$$f(t) \in \mathbb{V}_j \Leftrightarrow f(t - 2^j k) \in \mathbb{V}_j, \quad j, k \in \mathbb{Z}.$$

Further, let a family of such spaces $\mathbb{V}_j$ be nested such that $\mathbb{V}_{j+1} \subset \mathbb{V}_j$, so that $\lim_{j \to \infty} \mathbb{V}_j = \{0\}$ and $\lim_{j \to -\infty} \mathbb{V}_j = L^2(\mathbb{R})$. This nesting is done in such a way that the spaces are dilated versions of each other. More precisely,

$$f(t) \in \mathbb{V}_j \Leftrightarrow f\left(\frac{t}{2}\right) \in \mathbb{V}_{j+1}.$$

Each $\mathbb{V}_j$ is spanned by an orthonormal set of *scaling functions* $\phi_{j,k}$, which are obtained by translation and dilation of a certain function $\phi \in L^2(\mathbb{R})$ by

$$\phi_{j,k} := \frac{1}{\sqrt{2^j}} \phi\left(\frac{t - k}{2^j}\right), \quad j, k \in \mathbb{Z}.$$

Hence, due to orthonormality, the approximating projection of f is obtained as

$$P_{\mathbb{V}_j} f = \sum_{k=-\infty}^{+\infty} \langle f, \phi_{j,k} \rangle \phi_{j,k}.$$

The nesting of function spaces $\mathbb{V}_j \subset \mathbb{V}_{j-1}$ means that certain approximation details are lost between resolution levels, since any function in $\mathbb{V}_j$ can be expressed as a member of $\mathbb{V}_{j-1}$, but not the other way round. To capture those details, let $\mathbb{W}_j$ be the orthogonal complement of $\mathbb{V}_j$ with respect to $\mathbb{V}_{j-1}$, i. e.

$$\mathbb{V}_{j-1} = \mathbb{V}_j \oplus \mathbb{W}_j,$$

which implies

$$P_{\mathbb{V}_{j-1}} f = P_{\mathbb{V}_j} f + P_{\mathbb{W}_j} f.$$

This allows for the decomposition

$$L^2(\mathbb{R}) = \mathbb{V}_i \oplus \bigoplus_{j=-\infty}^{i} \mathbb{W}_j$$

for any $i \in \mathbb{Z}$, as well as a decomposition free of scaling functions,

$$L^2(\mathbb{R}) = \bigoplus_{j=-\infty}^{\infty} \mathbb{W}_j.$$

The detail spaces $\mathbb{W}_j$ are spanned by an orthonormal basis of *wavelet functions* $\psi_{j,k}$, which are derived from a *mother wavelet* $\psi \in L^2(\mathbb{R})$ by translation and dilations,

$$\psi_{j.k} := \frac{1}{\sqrt{2^j}} \psi\left(\frac{t - 2^j k}{2^j}\right).$$

For more details on the properties of ϕ and the derivation of ψ through *conjugate mirror filters*, see Mallat (2009).

Equipped with those definitions, any function $f \in L^2(\mathbb{R})$ can be expressed as a linear combination of scaling functions and wavelets,

$$f(t) = \sum_{k=-\infty}^{\infty} c_{i,k} \phi_{i,k}(t) + \sum_{j=-\infty}^{i} \sum_{k=-\infty}^{\infty} d_{j,k} \psi_{j,k}(t).$$

Here, $c_{i,k}$ are called *scale coefficients*, and $d_{j,k}$ are called *detail coefficients* or *wavelet coefficients*. They can be expressed as inner products

$$c_{j,k} = \left\langle \phi_{j,k}, f \right\rangle, \text{ and}$$

$$d_{j,k} = \left\langle \psi_{j,k}, f \right\rangle.$$

The set of those coefficients is called the *discrete wavelet transform (DWT)*.

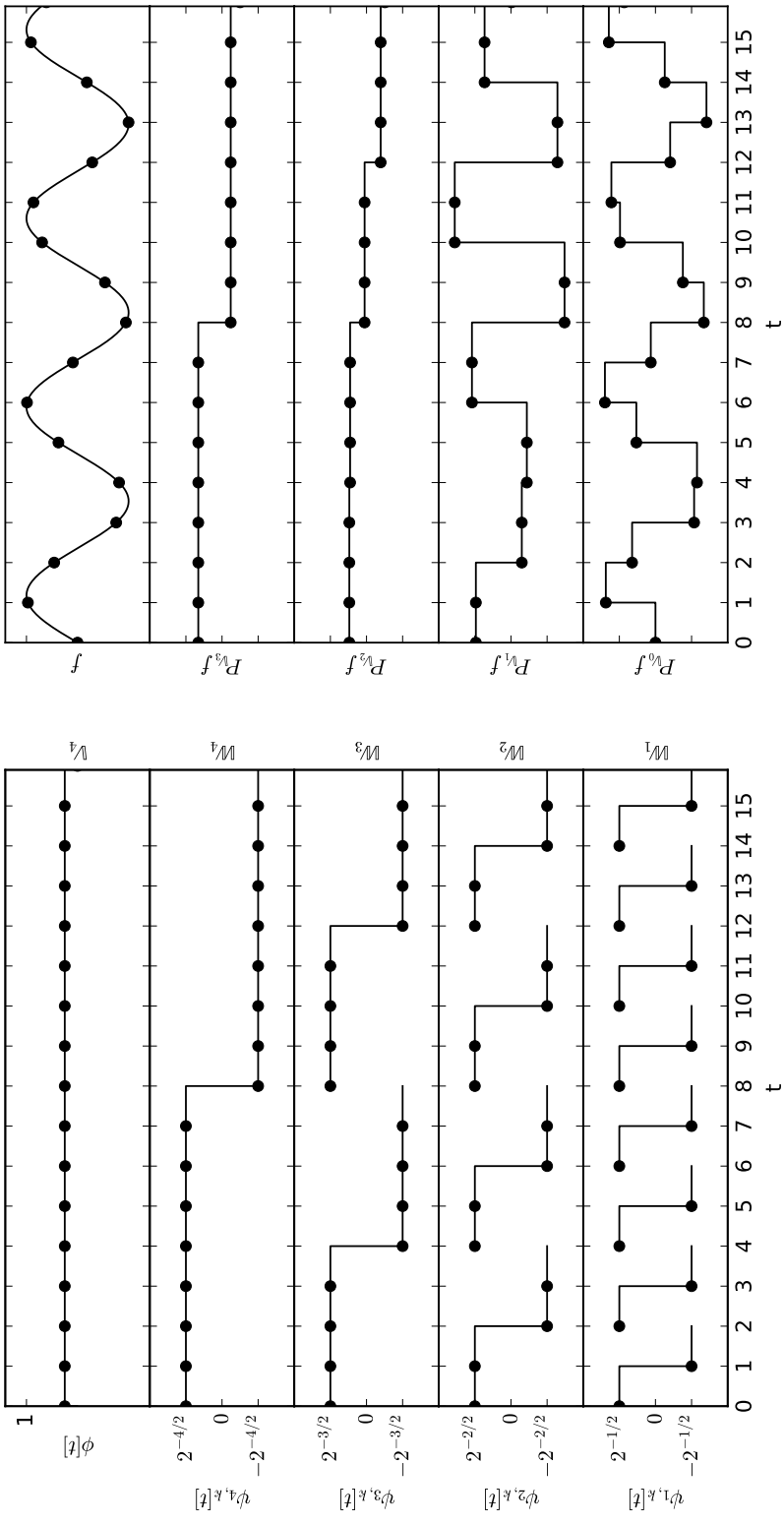


Figure 3.1: Left: The Haar wavelet basis, for a vector of size $T = 16$. Integer values obtained from discrete sampling are shown as bullets. The top subplot shows the scaling function $\phi[t]$, the plots below show the Haar wavelet functions at different resolution levels, with resolution increasing for decreasing j. These are the basis elements for detail spaces $\mathbb{W}_j$. Only the non-zero support is shown. At each level, k increases from left to right. The discretization $\boldsymbol{\psi}_{j,k}$ is marked by bullets, the lines represent the real-valued functions $\psi_{j,k}$. Each wavelet function has a central discontinuity $b^{\pm}_{j,k}$, which is connected here by a vertical line, as well as a left and right discontinuity $b^{+}_{j,k}$ and $b^{-}_{j,k}$. For instance, $b^{+}_{2,1} = 4$, $b^{\pm}_{2,1} = 6$, and $b^{-}_{2,1} = 8$. Notice the nested, tree-like structure of this basis. Right: Projections of $f = \sin\left(\frac{4}{3}t\right)$ (top) onto different approximation spaces $\mathbb{V}_j$. For each subplot on the right, the basis elements are those in the subplots at and above that level on the left, e. g. the basis for $\mathbb{V}_2$ comprises $\mathbb{W}_3$, $\mathbb{W}_4$, and $\mathbb{V}_4$. Notice that for integer coordinates $P_{\mathbb{V}_0}\boldsymbol{f} = \boldsymbol{f}$, i. e. the discrete wavelet transform (DWT) is lossless for equidistant samples $\boldsymbol{f}$ of f.

In computational applications, a physical signal f would often require the measurement, storage and processing of infinitely many values, unless the signal is discrete in nature. Instead, a vector $\boldsymbol{f}$ is obtained by sampling f at equidistant intervals. Let $\ell^2 \coloneqq L^2(\mathbb{N}, \mathcal{P}(\mathbb{N}), \#)$ be the L^2 space over the natural numbers with counting measure $\#$. It is also a separable Hilbert space, and can be treated as the discrete analogue of L^2, where

$$\langle \boldsymbol{f}, \boldsymbol{g} \rangle = \int_{\mathbb{N}} \langle \boldsymbol{f}[t], \boldsymbol{g}[t] \rangle \, d\#(t) = \sum_{t \in \mathbb{N}} \boldsymbol{f}[t] \, \boldsymbol{g}[t].$$

Notice that we prefer this notation to $\boldsymbol{f} \cdot \boldsymbol{g}$ to emphasize its connection to the function space. The discrete versions of the wavelets are then obtained as $\boldsymbol{\psi}_{j,k}[t] \coloneqq \psi_{j,k}(t)$ for $t \in \mathbb{N}$.

Further, assume finitely many sampling points such that $\boldsymbol{f} \in \mathbb{R}^T$. For the moment, assume T is a power of 2; data with $T \notin 2^{\mathbb{N}}$ can be treated as truncation of longer data, which can be generated using various padding methods (Strang & Nguyen 1997). The wavelet basis is then finite, and forms the rows of a matrix $\mathcal{W} \in \mathbb{R}^{T \times T}$, such that $\mathcal{W}\boldsymbol{y}$ yields the vector of wavelet coefficients. This is called the *discrete-time wavelet transform (DTWT)*, though DWT is often used, as any software implementation is by necessity discrete. Since the wavelet basis is orthonormal, $\mathcal{W}$ is orthogonal, i. e. $\mathcal{W}^{-1} = \mathcal{W}^\intercal$. Notice that the order of rows in $\mathcal{W}$ is unspecified, and hence different permutations of the coefficient vector are possible. Surprisingly, there are linear-time algorithms to calculate $\mathcal{W}\boldsymbol{y}$ (Sweldens 1995; Mallat 1989).

3.2 Wavelet regression

One of the main applications of the discrete-time wavelet transform is in functional regression, specifically in a method known as *wavelet thresholding* or *shrinkage*. Let the observed signal be a sampling of a function f corrupted by centered Gaussian noise, i. e.

$$\boldsymbol{y} = \boldsymbol{f} + \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon}[t] \sim_{\text{i.i.d.}} N(0, \sigma^2).$$

The goal of regression is to find an estimate $\widehat{\boldsymbol{f}}$ which approximates $\boldsymbol{f}$ using the available data $\boldsymbol{y}$. The quality of regression is usually discussed in the framework of decision theory. Before we discuss wavelet regression, we review some central concepts and results. The exposition follows Robert (2007) and Mallat (2009), with terminology following the former. An overview of relevant concepts is shown in Fig. 3.2.

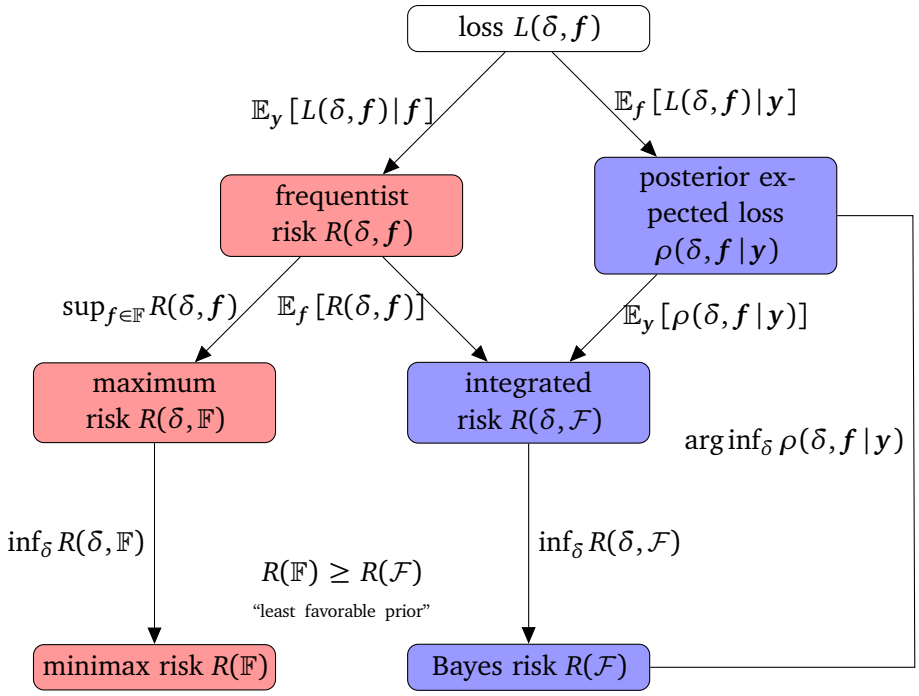


Figure 3.2: An overview of decision theory for regression. Concepts aligned with a frequentist framework are shown in red, and Bayesian in blue.

3.2.1 Decision theory for regression operators

Let $\mathbb{F}$ be a set of functions such that $f \in \mathbb{F}$, and $\mathcal{F}$ be a distribution over $\mathbb{F}$. Let δ be a decision operator, i. e. a regression method for estimating $\widehat{f} := \delta(\boldsymbol{y})$. Let $L(\delta(\boldsymbol{y}), f)$ be a non-negative *loss function* which quantifies the error for estimating f by $\widehat{f}$. The *frequentist risk*, or *risk* for short, quantifies the expected loss

$$R(\delta, f) := \mathbb{E}_{\boldsymbol{y}}[L(\delta(\boldsymbol{y}), f)\,|\,f] = \int L(\delta, f)\mathcal{P}(\boldsymbol{y}\,|\,f)\,\mathrm{d}\boldsymbol{y} \tag{3.1}$$

over all data instances for a given f.

In order to summarize the risk across all instances of f, there are two principle ways. If its prior distribution $\mathcal{F}$ is known, the *integrated risk* is obtained as the expectation with respect to $\mathcal{F}$ as

$$R(\delta, \mathcal{F}) := \mathbb{E}_f[R(\delta, f)] = \int R(\delta, f)\mathcal{P}(f)\,\mathrm{d}f\,. \tag{3.2}$$

The operator $\delta_{\mathcal{F}}$ achieving the *Bayes risk*[1]

$$R(\mathcal{F}) := R(\delta_{\mathcal{F}}, \mathcal{F}) = \inf_\delta R(\delta, \mathcal{F}) \tag{3.3}$$

[1]Naming conventions vary among authors: Mallat (2009) calls the posterior risk the *Bayes risk*, and that achieved by $\delta_{\mathcal{F}}$ is called the *minimum Bayes risk*.

is called a *Bayes estimator*. On the other hand, if such a prior is not known, the *maximum risk*

$$R(\delta, \mathbb{F}) := \sup_{\boldsymbol{f} \in \mathbb{F}} R(\delta, \boldsymbol{f}).$$

is the largest risk incurred by any $\boldsymbol{f}$ in $\mathbb{F}$. An operator $\delta_{\mathbb{F}}$ achieving the *minimax risk*

$$R(\mathbb{F}) := R(\delta_{\mathbb{F}}, \mathbb{F}) = \inf_{\delta} R(\delta, \mathbb{F}),$$

is called a *minimax estimator*. It is easy to show that the minimax risk is an upper bound to the Bayes risk, i. e.

$$R(\mathcal{F}) \leq R(\mathbb{F}),$$

since including prior information on the distribution of $\boldsymbol{f}$ can only improve the estimation. By the *minimax theorem* (Wald 1945; Komiya 1988; Sion 1958) minimax estimations are Bayes estimations for a *least favorable prior*, i. e. a prior distribution $\mathcal{F}_\circ$ achieving the minimax risk,

$$R(\mathbb{F}) = R(\mathcal{F}_\circ).$$

Though being a very conservative criterion, minimaxity tries to capture the notion of an *optimal* estimator. For instance, the sample mean is minimax for estimating the mean of i.i.d. Gaussian random variables, and maximum likelihood estimates for parametric settings are asymptotically locally minimax (see Donoho, Johnstone, et al. (1995) for a detailed discussion).

The question remains as to how a Bayes estimator can be derived. Notice that the above definitions are based on a frequentist interpretation of risk, as it is defined for a known $\boldsymbol{f}$ and integrated over all possible variates $\boldsymbol{y}$, even though $\boldsymbol{f}$ is latent and only one realization of $\boldsymbol{y}$ is observed. It is therefore more natural to take the Bayesian approach: the *posterior expected loss* or *posterior risk*

$$\rho(\delta, \boldsymbol{f} \,|\, \boldsymbol{y}) := \mathbb{E}_f \left[L(\delta, \boldsymbol{f}) \,|\, \boldsymbol{y} \right] = \int L(\delta, \boldsymbol{f}) \mathcal{P}(\boldsymbol{f} \,|\, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{f} \tag{3.4}$$

quantifies the expected loss for the given data $\boldsymbol{y}$ across all true functions $\boldsymbol{f} \sim \mathcal{F}$ with respect to a prior distribution $\mathcal{F}$. The Bayes estimator can then be obtained by selecting the optimal estimator for each $\boldsymbol{y}$ individually as

$$\delta_{\mathcal{F}}(\boldsymbol{y}) := \arg\inf_{\delta} \rho(\delta, \boldsymbol{f} \,|\, \boldsymbol{y}),$$

since

$$R(\delta, \mathcal{F}) = \int R(\delta, \boldsymbol{f}) \mathcal{P}(\boldsymbol{f}) \, \mathrm{d}\boldsymbol{f} = \int \rho(\delta, \boldsymbol{f} \,|\, \boldsymbol{y}) \mathcal{P}(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} \,. \tag{3.5}$$

This is of particular interest if the loss function is quadratic, i. e.

$$L(\delta(\boldsymbol{y}), \boldsymbol{f}) = \|\delta(\boldsymbol{y}) - \boldsymbol{f}\|^2.$$

In this case, a Bayes estimator for each $\boldsymbol{y}$ corresponds to the mean of the posterior distribution, the *posterior expectation*

$$\delta_{\mathcal{F}}(\boldsymbol{y}) = \mathbb{E}\,[\boldsymbol{f} \,|\, \boldsymbol{y}] = \int \boldsymbol{f} \mathcal{P}(\boldsymbol{f} \,|\, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{f}. \tag{3.6}$$

Expectation holds component-wise,

$$\forall t: \quad \widehat{\boldsymbol{f}}[t] = \mathbb{E}\,[\boldsymbol{f}[t] \,|\, \boldsymbol{y}], \tag{3.7}$$

see Mallat (2009, Theorem 11.1). This fact will be used later in this thesis to connect Hidden Markov Models to wavelet regression. Other loss functions yield different statistics of the posterior distribution. For instance, under certain conditions (Bassett & Deride 2018), the Bayes estimator for the *0-1 loss*

$$L_\epsilon(\delta, \boldsymbol{f}) := \begin{cases} 0 & \|\boldsymbol{f} - \boldsymbol{f}\| < \epsilon \\ 1 & \left\|\widehat{\boldsymbol{f}} - \boldsymbol{f}\right\| \geq \epsilon \end{cases},$$

$\epsilon > 0$, converges to the *posterior mode*, more commonly called the *maximum a posteriori (MAP) estimate*, for $\epsilon \to 0$.

3.2.2 Minimaxity of wavelet thresholding

In regression settings, it is often impossible in practice to formulate a prior signal distribution $\mathcal{F}$, and the Bayes estimator (Eq. (3.7)) is generally unattainable. For instance, no practical statistical model exists to accurately describe photographic images. Therefore, other criteria, such as minimaxity with respect to a signal class $\mathbb{F}$, are usually employed to select a regression method (*admissibility* being another popular criterion). Unfortunately, deriving minimax estimators is often hard, and they tend to be hard to compute. However, good approximations can sometimes be obtained. Wavelets in particular allow for a very simple approximation of minimaxity.

Consider a *diagonal estimator* δ in which each detail coefficient $d_{j,k}$ is attenuated by a factor $a_{j,k}$, which depends on $d_{j,k}$ itself. The risk is then

$$R(\delta,\boldsymbol{f}) = \mathbb{E}\left[\|\boldsymbol{f}-\delta(\boldsymbol{y})\|^2\right] = \sum_{j,k}\mathbb{E}\left[\left|\left\langle\boldsymbol{\psi}_{j,k},\boldsymbol{f}\right\rangle - a_{j,k}\left\langle\boldsymbol{\psi}_{j,k},\boldsymbol{Y}\right\rangle\right|^2\right]$$

Obviously, this attenuation requires knowledge of $\boldsymbol{f}$, and is therefore not attainable, unless an oracle for $\boldsymbol{f}$ is provided. For theoretical purposes, consider the existence of an oracle having knowledge of $\boldsymbol{f}$. The *oracle attenuator* $\delta^{\boldsymbol{f}}_{\text{att}}$ minimizes the risk by setting

$$a_{j,k} = \frac{\left|\left\langle\boldsymbol{\psi}_{j,k},\boldsymbol{f}\right\rangle\right|^2}{\left|\left\langle\boldsymbol{\psi}_{j,k},\boldsymbol{f}\right\rangle\right|^2+\sigma^2}.$$

Restricting the attenuation coefficients to $\{0,1\}$ yields the *oracle projector* $\delta^{\boldsymbol{f}}_{\text{pr}}$ for which the best risk is obtained by

$$a_{j,k} = \begin{cases} 1 & \left|\left\langle\boldsymbol{\psi}_{j,k},\boldsymbol{f}\right\rangle\right| \geq \sigma \\ 0 & \left|\left\langle\boldsymbol{\psi}_{j,k},\boldsymbol{f}\right\rangle\right| < \sigma \end{cases}. \tag{3.8}$$

In Donoho & Johnstone (1994), this method is referred to as *selective wavelet reconstruction*. It can be shown that the oracle projection risk comes close to the optimal attenuation, since

$$\frac{1}{2}R\left(\delta^{\boldsymbol{f}}_{\text{pr}},\boldsymbol{f}\right) \leq R\left(\delta^{\boldsymbol{f}}_{\text{att}},\boldsymbol{f}\right) \leq R\left(\delta^{\boldsymbol{f}}_{\text{pr}},\boldsymbol{f}\right),$$

and hence

$$R\left(\delta^{\boldsymbol{f}}_{\text{pr}},\boldsymbol{f}\right) \leq 2R\left(\delta^{\boldsymbol{f}}_{\text{att}},\boldsymbol{f}\right).$$

Though the oracle is inaccessible, oracle estimators can be approximated surprisingly well. Let the *universal threshold* be

$$\lambda_u := \sqrt{2\ln T}\sigma. \tag{3.9}$$

The *hard-thresholding estimator* δ_{th} is a projector using

$$a_{j,k} = \begin{cases} 1 & \left|\left\langle\boldsymbol{\psi}_{j,k},\boldsymbol{y}\right\rangle\right| \geq \lambda_u \\ 0 & \left|\left\langle\boldsymbol{\psi}_{j,k},\boldsymbol{y}\right\rangle\right| < \lambda_u \end{cases} \tag{3.10}$$

which achieves a risk of

$$R(\delta_{\text{th}},\boldsymbol{f}) \leq (2\ln T+1)\left(\sigma^2 + R\left(\delta^{\boldsymbol{f}}_{\text{pr}},\boldsymbol{f}\right)\right) \tag{3.11}$$

for $T \geq 4$; see Mallat (2009, Theorem 11.7) for details. δ_{th} is optimal in the sense that the risk cannot be improved by any other diagonal estimator. It is also asymptotically minimax over a wide range of smoothness classes (Donoho & Johnstone 1995). Most importantly for this thesis, it is asymptotically minimax over piecewise α-Lipschitz functions, including the case where $\mathbb{F}$ is the set of piecewise-constant functions.

Using universal thresholding for wavelet regression, also called *wavelet shrinkage*, has a very intuitive explanation: Following Donoho & Johnstone (1994), a wavelet is said to have m *vanishing moments* if

$$\left\langle p^i, \psi \right\rangle = 0, \quad 0 \leq i < m, p \text{ scalar.}$$

It follows that ψ is orthogonal to any polynomial f of degree less than m, since

$$\left\langle \sum_{i=1}^{m-1} p^i, \psi \right\rangle = \sum_{i=1}^{m-1} \left\langle p^i, \psi \right\rangle = 0.$$

This property is called *polynomial suppression* (Mallat 2009). It follows that the detail coefficients in any wavelet decomposition of such a polynomial f are all zero, as it can be expressed entirely in terms of the scaling functions. Furthermore, any function in C^m is well approximated by a $(m-1)$-degree Taylor polynomial p_v about point v over a finite interval $[v-h, v+h]$. Let $f = p_v + \epsilon_v$. Then

$$\left\langle f, \psi_{j,k} \right\rangle = \left\langle \epsilon_v, \psi_{j,k} \right\rangle,$$

so the wavelet coefficient is negligible, as it only measures the Taylor approximation error, which is bounded by the Lipschitz coefficient of f as

$$|\epsilon_v(t)| \leq K|t-v|^\alpha, \quad K > 0, \quad \text{and } m = \lfloor \alpha \rfloor < n.$$

Since the wavelet transform is linear, it acts on the signal and noise component independently, i. e.

$$\mathcal{W}\boldsymbol{y} = \mathcal{W}(\boldsymbol{f} + \boldsymbol{\epsilon}) = \mathcal{W}\boldsymbol{f} + \mathcal{W}\boldsymbol{\epsilon}.$$

The central idea in wavelet shrinkage is that the detail coefficients $d_{j,k} = \left\langle \boldsymbol{f}, \boldsymbol{\psi}_{j,k} \right\rangle$ are 0 if $\boldsymbol{f}$ is polynomial over the entire support of $\boldsymbol{\psi}_{j,k}$ due to polynomial suppression, or vanishingly small if f is locally m-times differentiable. Furthermore, due to orthogonality of $\mathcal{W}$, $\mathcal{W}\boldsymbol{\epsilon}$ is again a

random vector of i.i.d. random variables distributed as $N(0, \sigma^2)$, so the noise is maintained under the wavelet transform. In general, orthogonal maps preserve the L^2 norm, so

$$\|\mathcal{W}\boldsymbol{\epsilon}\| = \|\boldsymbol{\epsilon}\| \quad \text{and} \quad \|\mathcal{W}\boldsymbol{y}\| = \|\boldsymbol{y}\|.$$

It follows that for piecewise polynomial functions with only a few discontinuities, most signal coefficients $\langle \boldsymbol{f}, \boldsymbol{\psi}_{j,k} \rangle$ are zero due to polynomial suppression, hence

$$\langle \boldsymbol{y}, \boldsymbol{\psi}_{j,k} \rangle = \langle \boldsymbol{\epsilon}, \boldsymbol{\psi}_{j,k} \rangle$$

for most j,k, i. e. most wavelet coefficients arise entirely from noise. As the noise component of the data is preserved as a Gaussian vector of variance σ^2 under the orthogonal transformation $\mathcal{W}$, most of the T detail coefficients are themselves i.i.d. Gaussian random variables. The idea is then to find a way to create a vector $\boldsymbol{w}$ by setting a suitable set of coefficients in $\mathcal{W}\boldsymbol{f}$ to zero, and then use the inverse wavelet transform as a regression $\widehat{\boldsymbol{f}} \coloneqq \mathcal{W}^\intercal \boldsymbol{w}$. Ideally, these would be those exactly those noise coefficients. As a filtering criterion, the universal threshold can thus be interpreted as the expected maximum deviation of T such Gaussian random variables, which is at most $\lambda_u \coloneqq \sqrt{2 \ln T}\sigma$ by Cramér-Chernoff's method (Massart 2003). A simple derivation of this result has appeared multiple times in the mathematical vernacular:

Proposition 3.2.1 (Expected maximum of Gaussian random variables)**.** *Let $(X_i)_{i=1}^T$ a sequence of T centered i.i.d Gaussian random variables with variance σ^2. Let $Z \coloneqq \max_i X_i$ Then,*

$$\mathbb{E}[Z] \leq \sqrt{2 \ln T}\sigma$$

Proof. Using Jensen's inequality,

$$\exp(t\mathbb{E}[Z]) \leq \mathbb{E}[\exp(tZ)] = \mathbb{E}\left[\exp\left(t \max_i X_i\right)\right] = \mathbb{E}\left[\max_i \exp(tX_i)\right] \leq \sum_{i=1}^{T} \mathbb{E}[\exp(tX_i)].$$

The last term can be expressed a product of Gaussian moment-generating functions

$$\exp\left(t\mu + t^2 \frac{\sigma^2}{2}\right)$$

with $\mu_i = 0$, hence

$$\exp(t\mathbb{E}[Z]) \leq T \exp\left(t^2 \frac{\sigma^2}{2}\right)$$

and

$$\mathbb{E}[Z] \leq \frac{\ln T}{t} + t\frac{\sigma^2}{2}$$

The minimum is attained at $t = \frac{\sqrt{2\ln T}}{\sigma}$, and the claim follows. □

The same upper bound can be shown for $Z := \max_i |X_i|$, though this expectation will generally be larger. Furthermore, the maximum noise coefficient is just below λ_u with high probability (Berman 1992):

$$\lim_{T\to\infty} \mathbb{P}\left(\max_{0\leq i<T} \left|\left\langle \boldsymbol{\psi}_{j,k}, \boldsymbol{\epsilon}\right\rangle\right| \in \left[\lambda_u - \sigma\frac{\ln\ln T}{\ln T}, \lambda_u\right]\right) = 1.$$

3.3 The Haar wavelet

The HMM treated in this thesis generate piecewise constant data with noise, albeit with potentially different noise variances for each state. As the underlying sequence of means is piecewise linear, we can base our method on the simplest, piecewise constant form of wavelets. This decomposition goes back over a century to the work of Haar (1910), and is widely recognized as the first example of a wavelet transform, long before the broader theory was established. Let the Haar scaling function be

$$\phi(t) := \begin{cases} 1 & 0 \leq t < 1 \\ 0 & \text{elsewhere.} \end{cases}$$

The *Haar wavelet* constructed from that scaling function is

$$\psi(t) := \begin{cases} 1 & 0 \leq t < \frac{1}{2} \\ -1 & \frac{1}{2} \leq t < 1 \\ 0 & \text{elsewhere.} \end{cases}$$

As before, the basis elements are defined as

$$\psi_{j,k} := \frac{1}{\sqrt{2^j}}\psi\left(\frac{t - 2^j k}{2^j}\right),$$

where $\boldsymbol{\psi}_{j,k}$ has non-zero support over the interval $[2^j k, 2^j(k+1))$. The Haar wavelet has one vanishing moment, so in general it is orthogonal only to constant functions.

The discrete-time Haar wavelet transform $\mathcal{W}\boldsymbol{y}$ for a vector $\boldsymbol{y}$ of size T consists of the scaling coefficient $c_{0,0}$ as well as the detail coefficients $d_{j,k}$, $j \in \{1, \dots, \operatorname{ld} T\}$, $k \in \left\{0, \dots, \frac{T}{2^j} - 1\right\}$. The rows of $\mathcal{W}$ consist of all $\boldsymbol{\psi}_{j,k}$ as well as $\boldsymbol{\phi}$. Let

$$b^+_{j,k} \coloneqq 2^j k, \qquad b^\pm_{j,k} \coloneqq 2^j \left(k + \frac{1}{2}\right), \text{ and} \qquad b^-_{j,k} \coloneqq 2^j (k+1)$$

be the position after the left, central and right discontinuity, respectively. Let

$$L_{j,k} \coloneqq \left\{b^+_{j,k}, \dots, b^\pm_{j,k} - 1\right\} \qquad R_{j,k} \coloneqq \left\{b^\pm_{j,k}, \dots, b^-_{j,k} - 1\right\}$$

denote the index sets of the left (positive) and right (negative) support interval of $\boldsymbol{\psi}_{j,k}$, i. e. $\boldsymbol{\psi}_{j,k}[t] > 0$ for $t \in L_{j,k}$ and $\boldsymbol{\psi}_{j,k}[t] < 0$ for $t \in R_{j,k}$. Note that $\boldsymbol{\psi}_{j,k}$ is zero outside of these index sets, so we ignore those entries and only refer to $L \cup R$ as the support of the wavelet. The wavelet basis is illustrated in Fig. 3.1.

W.l.o.g., let $k = 0$, $\boldsymbol{\psi}_j \coloneqq \boldsymbol{\psi}_{j,0}$, $L_j \coloneqq L_{j,0}$, and $R_j \coloneqq R_{j,0}$. Recall that for $X_i \sim N(\mu_i, \sigma_i^2)$ and $s_i \in \{-1, 1\}$, the term

$$\sum_i s_i a_i X_i \sim N\left(\sum_i s_i \mu_i, \sum_i a_i^2 \sigma^2\right),$$

hence

$$\left\langle \boldsymbol{\epsilon}, \boldsymbol{\psi}_j \right\rangle = \frac{1}{\sqrt{2^j}} \left(\sum_{t \in L_j} \boldsymbol{\epsilon}[t] - \sum_{t \in R_j} \boldsymbol{\epsilon}[t]\right) \sim N\left(0, \sigma^2\right).$$

As expected, the noise is preserved under the Haar wavelet transform.

Chapter 4

Haar Wavelet Compression

In this chapter, we introduce the core methodological contribution of this thesis, using the established theory introduced in the previous chapters. We show that the expectation of posterior marginal state means in homoscedastic Gaussian Hidden Markov Model (HMM) is a Bayes estimator under prior parameters τ, and its risk is limited by the minimax risk of Haar wavelet regression. Using the positions of discontinuities as boundaries, blocks created by Haar wavelet compression preserve the approximate positions of true state transitions, and the probability of under-compression decreases with the separation of state means. Furthermore, we derive a dynamic compression scheme for heteroscedastic HMM, and implement it using a *wavelet tree* data structure. We provide an evaluation of our method on simulated and biomedical data, showing significant improvements in speed and convergence behavior over uncompressed Forward-Backward Gibbs sampling (FBG). Finally, we derive bounds on the bias incurred on the forward variables by compression for general HMM, and show that this tends to bias the forward-backward sampling procedure towards the maximum posterior marginal state within each compression block. We conjecture that our software HaMMLET produces an approximation of the *maximum posterior margins (MPM) segmentation* (see Section 2.4).

The central approach of this thesis is to integrate Haar wavelet regression with Forward-Backward Gibbs sampling, in order to derive approximate state marginals and Bayesian smoothing for Hidden Markov models. Significant speedup and memory savings can be obtained by data compression in which the discontinuities in the piecewise constant function $\widehat{\boldsymbol{f}}$, obtained from Haar wavelet regression towards the sequence $\boldsymbol{f}$ of emission means of the true state sequence $\boldsymbol{q}$,

are used to define boundaries for data compression blocks:

Definition 4.0.1 (Haar wavelet compression)**.** *Let $\boldsymbol{y}$ be a vector of input values, and $\boldsymbol{d}$ some vector of Haar wavelet coefficients of the same dimensionality (typically the result of some regression method on $\mathcal{W}\boldsymbol{y}$, such as wavelet shrinkage). Let $\boldsymbol{y}$ be partitioned into* blocks *of sufficient statistics such that a block starts at position t if and only if there exists some j,k such that $d_{j,k} \neq 0$ and at least one of $b^{+}_{j,k}$, $b^{\pm}_{j,k}$ or $b^{-}_{j,k}$ is equal to t (see p. 43). In other words, the data is compressed into blocks defined by the discontinuities in $\mathcal{W}^{\intercal}\boldsymbol{d}$.*

Notice that only projection operators, i. e. methods which set coefficients to zero, yield compression into blocks of at least two position, since any non-zero coefficient d_{ij} will introduce a discontinuity, and hence a compression block boundary $b^{\pm}_{ij}$, and in many cases at is left and right discontinuities b^{+}_{ij} and b^{-}_{ij} as well. Therefore, only projectors should be considered in a compression setting, and among them, thresholding methods achieve the best risk, with the universal threshold approaching the minimax risk.

4.1 Homoscedastic HMM

This proposal is based on the observation that HMM inference and regression methods deal with similar input data. Consider our observed data $\boldsymbol{y}$ to consist of a piecewise-constant data vector of means, $\boldsymbol{f}$, corrupted by additive noise $\boldsymbol{\epsilon}$, which is typically Gaussian. In a regression setting, the noise is considered to be homoscedastic, i. e. of uniform variance, while the number of distinct values in $\boldsymbol{f}$ is not limited a priori. On the other hand, for data generated by a Gaussian HMM, the number of distinct means is limited by the number of states, while there may be different noise variances associated with each component. The obvious intersection case is data generated by an HMM which has the same finite emission variance in each component, which yields a homoscedastic random vector $\boldsymbol{y}$:

Definition 4.1.1 (Homoscedastic HMM (σ-HMM))**.** *We call a Gaussian Hidden Markov model a* homoscedastic HMM *of variance σ^2, or σ-HMM, iff $\boldsymbol{\theta}_i = (\mu_i, \sigma^2)$. State labels can be identified with the emission means, so that the state sequence $\boldsymbol{q} \in \{1,2,\dots\}^T$ can be written as the discrete*

sampling

$$\boldsymbol{f} \coloneqq (\mu_{\boldsymbol{q}[1]}, \ldots, \mu_{\boldsymbol{q}[T]}) \in \{\mu_1, \mu_2, \ldots\}^T$$

of a piecewise constant function $f \in \mathbb{F}$ *such that*

$$\forall t \in \{1, \ldots, T\} : \boldsymbol{f}[t] = f(f), \textit{ and}$$

$$\forall x \in [t, t+1) : f(x) \coloneqq \boldsymbol{f}[t].$$

A σ*-HMM thus defines a probability distribution over* $\mathbb{F}$.

Since we can easily convert between f and $\boldsymbol{f}$, and minimaxity results for wavelet regression apply to discrete samplings $\boldsymbol{f}$ of f, we forgo the distinction between $\mathbb{R}^T$ and $\mathbb{F}$, and we refer to the vector $\boldsymbol{f}$ as a piecewise-constant function. Hence, on the one hand, data generated by a σ-HMM can be treated as a piecewise constant function with i.i.d. Gaussian noise, where $\boldsymbol{f}$ can be approximated by regression. On the other hand, $\boldsymbol{f}$ is a hidden Markov chain which has an explicit distribution given the HMM parameters, allowing for an explicit formulation of Bayes risk. Thus, the observed data $\boldsymbol{y}$ can be treated both within a regression as well as an HMM framework, allowing for an integration of wavelets and FBG.

As $\boldsymbol{f}$ is piecewise constant, the Haar wavelet with its one vanishing moment is the obvious candidate for regression. The emission variance σ^2 can be estimated directly using the standard approach of taking the median absolute value of finest detail coefficients $\left|d_{1,k}\right|$, see Mallat (2009, p. 565) for details. Having the smallest support of 2, very few wavelets will contain a discontinuity in $\boldsymbol{f}$ in their support. Since $d_{1,k} \sim N(0, \sigma^2)$, this corresponds to the median absolute deviation from $\mu = 0$, and hence

$$\widehat{\sigma}^2_{\text{MAD}} \coloneqq \left(\frac{\text{med}_k \left|d_{1,k}\right|}{\Phi^{-1}\left(\frac{3}{4}\right)} \right)^2 = \left(0.6745 \operatorname*{med}_k \left|d_{1,k}\right| \right)^2,$$

where Φ is the PDF of the standard normal distribution $N(0,1)$. Using the universal threshold $\sqrt{2 \ln T}\sigma_{\text{MAD}}$ (Donoho & Johnstone 1994), with high probability, the resulting estimate $\widehat{\boldsymbol{f}}_{\text{th}}$ will set to zero all coefficients of wavelets with support over ranges that do not contain state transitions, and hence have no discontinuity in $\boldsymbol{f}$. Conversely, it will keep the coefficients of those wavelets which have state transitions within their support. This creates discontinuities in the regression result around state transitions. We will develop the connection to HMM in the sections below.

4.2 Connection between HMM and wavelets

In this section, we relate wavelet shrinkage to HMM smoothing. Since σ-HMM define distributions over $\mathbb{F}$, we define two Bayes estimators based on the posterior state marginals. Under quadratic loss, these are obtained by position-wise posterior means, conditioned on the observed data $\boldsymbol{y}$. We argue that changes in one of these estimators reflect strong changes in the posterior state distribution. Since minimax estimators are Bayes estimators for least favorable priors, Haar wavelet regression appears as a limiting case, and block boundaries can be expected to occur around changes in the marginal state distribution. We also discuss how wavelet compression might bias the Gibbs sampler towards the posterior parameters, yielding faster convergence.

Definition 4.2.1 (Smoothing estimator)**.** *Let H be a σ-HMM with the hidden Markov chain*

$$\mathcal{M}(\boldsymbol{f}) \coloneqq \mathcal{P}(\boldsymbol{f} \mid \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi})$$

over the state space $\{\mu_i\}$. Let

$$\gamma_j(t) \coloneqq \mathcal{P}(\boldsymbol{q}[t] = j \mid \boldsymbol{y}, \boldsymbol{\theta}, \mathcal{A}, \boldsymbol{\pi})$$

be the marginal probability to be in state j at position t, as obtained by HMM smoothing via the forward-backward algorithm (Section 2.4.2). Then we call the estimator $\widehat{\boldsymbol{f}}_{\mathcal{M}} \coloneqq \delta_{\mathcal{M}}(\boldsymbol{y})$ with

$$\widehat{\boldsymbol{f}}_{\mathcal{M}}[t] \coloneqq \sum_i \gamma_i(t)\,\mu_i$$

the smoothing estimator *for $\boldsymbol{f}$ with respect to H.*

An example for this estimator is shown in Fig. 4.1. Notice how $\widehat{\boldsymbol{f}}_{\mathcal{M}}$ provides a summary of the behavior of the marginal state distributions, and can therefore be used as an indicator of the loci at which the marginal state probabilities change:

Proposition 4.2.1. *Given data $\boldsymbol{y}$, the change in the smoothing estimator between positions t and u is*

$$\widehat{\boldsymbol{f}}_{\mathcal{M}}[t] - \widehat{\boldsymbol{f}}_{\mathcal{M}}[u] = \sum_{j=1}^{k-1} \big(\gamma_j(t) - \gamma_j(u)\big)(\mu_j - \mu_k),$$

and that difference between any two positions scales linearly with the separation of means.

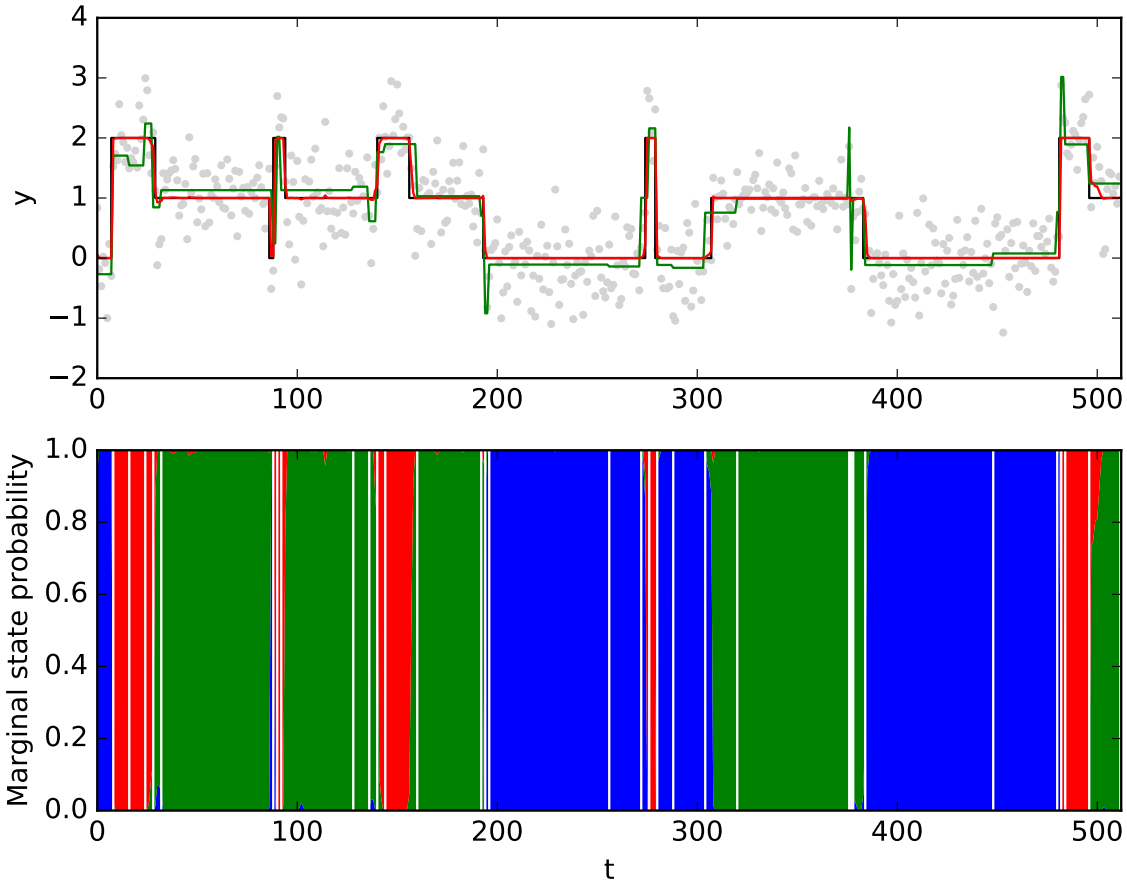


Figure 4.1: Smoothing estimator $\widehat{f}_{\mathcal{M}}(y)$ (top figure, red curve) for a state sequence f (black curve), and emission values y (gray dots) generated by a 3-state σ-HMM with known parameters. Wavelet thresholding $\widehat{f}_{\text{th}}$ (green curve) is the Bayes estimator that minimizes the risk incurred under the least favorable parametrization of a σ-HMM. The bottom figure shows the true marginal state distributions at each position. White vertical lines indicate the position of discontinuities in $\widehat{f}_{\text{th}}$ and hence the compression block boundaries.

Proof. Let $d_j = \gamma_j(u) - \gamma_j(t)$ be the change in marginal state probability for state j, and $\gamma_j := \gamma_j(t)$. Since $\sum_{j=1}^{k} \gamma_j(t) = \sum_{j=1}^{k} \gamma_j(u) = 1$, we have

$$\widehat{f}_{\mathcal{M}}[t] - \widehat{f}_{\mathcal{M}}[u] =$$

$$\left(\sum_{j=1}^{k-1} \gamma_j \mu_j + \left(1 - \sum_{j=1}^{k-1} \gamma_j\right)\mu_k\right) - \left(\sum_{j=1}^{k-1} (\gamma_j + d_j)\mu_j + \left(1 - \left(\sum_{j=1}^{k-1} \gamma_j\right) - \left(\sum_{j=1}^{k-1} d_j\right)\right)\mu_k\right)$$

$$= \left(\sum_{j=1}^{k-1} \gamma_j \mu_j\right) + \mu_k - \left(\sum_{j=1}^{k-1} \gamma_j \mu_k\right) - \left(\sum_{j=1}^{k-1} \gamma_j \mu_j\right) - \left(\sum_{j=1}^{k-1} d_j \mu_j\right) - \mu_k + \left(\sum_{j=1}^{k-1} \gamma_j \mu_k\right) + \left(\sum_{j=1}^{k-1} d_j \mu_k\right)$$

$$= \left(\sum_{j=1}^{k-1} d_j \mu_k\right) - \left(\sum_{j=1}^{k-1} d_j \mu_j\right) = \sum_{j=1}^{k-1} d_j (\mu_k - \mu_j).$$

Further, w.l.o.g. let $\mu_1 \leq \mu_2 \leq \cdots \leq \mu_k$. Then $\mu_k - \mu_j = \left|\mu_k - \mu_j\right|$ for $k > j$, and for any scaling factor $a \in \mathbb{R}$ of mean separation,

$$\sum_{j=1}^{k-1} d_j a \left|\mu_k - \mu_j\right| = a \sum_{j=1}^{k-1} d_j \left|\mu_k - \mu_j\right| = a\left(\widehat{f}_{\mathcal{M}}[t] - \widehat{f}_{\mathcal{M}}[u]\right).$$

□

Notice that shifts in $\widehat{f}_{\mathcal{M}}$ are small if means are not well separated, changes in marginal state probabilities are small, or in rare cases, the vector of changes d_j is orthogonal to that of means μ_j. Conversely, large mean separation or large state probability changes increase shifts in $\widehat{f}_{\mathcal{M}}$.

Having obtained a regression operator which captures significant changes in the marginal state distribution, we now relate it to wavelet regression using the framework of Bayesian decision theory:

Proposition 4.2.2. *$\delta_{\mathcal{M}}$ is a Bayes estimator for $\boldsymbol{f}$ with respect to prior $\mathcal{P}(\boldsymbol{f} \mid \mathcal{A}, \boldsymbol{\pi}, \boldsymbol{\theta})$ under quadratic loss $\|\boldsymbol{f} - \delta_{\mathcal{M}}(\boldsymbol{y})\|^2$.*

Proof. For a quadratic loss function, any Bayes estimator equals the mean of the posterior distribution given $\boldsymbol{y}$ (Eq. (3.7)). Conditioning the hidden Markov chain $\mathcal{M}(\boldsymbol{f})$ on the observed data $\boldsymbol{y}$ yields the posterior

$$\mathcal{M}(\boldsymbol{f} \mid \boldsymbol{y}) = \mathcal{P}(\boldsymbol{f} \mid \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{y}),$$

i. e. the state sequence distribution in the HMM for which $\mathcal{M}$ is the latent state process. Then,

$$\widehat{f}_{\mathcal{M}}[t] = \mathbb{E}\left[\boldsymbol{f}[t] \mid \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{y}\right] = \int \boldsymbol{f}[t] \mathcal{P}(\boldsymbol{f} \mid \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{f}[t],$$

and therefore

$$\widehat{f}_{\mathcal{M}}[t] = \sum_j \mu_j \mathcal{P}(\boldsymbol{q}[t] = j \mid \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{y}) = \sum_j \mu_j \gamma_j(t)$$

attains the Bayes risk $R(\mathcal{M})$. □

While a σ-HMM plays the role of a prior distribution on $\mathbb{F}$ in the decision-theoretic sense, we can extend the argument to Bayesian σ-HMM with a true prior for the model:

Proposition 4.2.3 (HMM estimator)**.** *Let $\delta_\tau(\boldsymbol{y})$ with*

$$\widehat{f}_\tau[t] \coloneqq \mathbb{E}\left[\boldsymbol{f}[t] \mid \boldsymbol{y}, \boldsymbol{\tau}\right]$$

be called HMM estimator *for* $\boldsymbol{f}$. *This is a Bayes estimator for a prior distribution* $\mathcal{F} = \mathcal{P}(\boldsymbol{f} \mid \boldsymbol{y}, \boldsymbol{\tau})$ *over the set of piecewise constant functions* $\mathbb{F}$ *under quadratic loss. Its risk under a least favorable parametrization* $\mathcal{M}_\circ := \mathcal{M}(\boldsymbol{f} \mid \boldsymbol{\tau}_\circ)$ *is bounded by the minimax risk over* $\mathbb{F}$*, asymptotically attained by Haar wavelet regression,*

$$R(\mathcal{M}_\circ) \leq R(\mathbb{F}).$$

Proof. This follows directly from the results in Section 3.2. Since the parameters to $\mathcal{M}$ themselves are subject to priors of their own, it is obvious that $\mathcal{P}(\boldsymbol{f} \mid \boldsymbol{y}, \boldsymbol{\tau})$, analogous to $\mathcal{P}(\boldsymbol{q} \mid \boldsymbol{y}, \boldsymbol{\tau})$ (Eq. (2.20)), defines a prior distribution over $\mathbb{F}$ in the decision-theoretic sense, with all prior information about the HMM encoded in $\boldsymbol{\tau}$. Under quadratic loss, the Bayes estimator for each $\boldsymbol{y}$ is obtained as

$$\int \boldsymbol{f}[t] \mathcal{P}(\boldsymbol{f}[t] \mid \boldsymbol{y}, \boldsymbol{\tau}) \, \mathrm{d}\boldsymbol{f}[t] = \mathbb{E}\left[\boldsymbol{f}[t] \mid \boldsymbol{y}, \boldsymbol{\tau}\right].$$

□

Notice that this estimator is hard to compute for the same reasons that we use MCMC to sample the HMM posterior in the first place. It only serves to justify the use of Haar wavelet compression. However, in analogy to Eq. (2.22), it could be easily approximated using Forward-Backward Gibbs sampling, as

$$\widehat{\boldsymbol{f}}_\tau[t] \approx \frac{1}{N} \sum_{i=1}^{N} \mu^{(i)}_{\boldsymbol{q}[t]}.$$

Since both estimators $\widehat{\boldsymbol{f}}_{\mathrm{th}}$ and $\widehat{\boldsymbol{f}}_\tau$ are bounded by the minimax risk, and wavelet thresholding typically yields a good approximation of $\boldsymbol{f}$, we expect $\|\widehat{\boldsymbol{f}}_{\mathrm{th}} - \widehat{\boldsymbol{f}}_\tau\|$ to be small. Consequently, we expect the discontinuities of the wavelet regression $\widehat{\boldsymbol{f}}_{\mathrm{th}}$ to roughly correspond to changes in the posterior state distribution of a Bayesian HMM. Hence, by virtue of being a minimax estimator of $\boldsymbol{f}$, wavelet shrinkage introduces information about the true posterior state marginals into the sampling process before the posterior distribution is ever available to the sampler. We believe this makes for a strong case to use those discontinuities as block boundaries for compression.

FBG samples from $\mathcal{P}(\boldsymbol{f} \mid \boldsymbol{y}, \boldsymbol{\tau})$ only once the underlying Markov chain has converged. While we argue that wavelet compression is *compatible* with a σ-HMM in the sense that changes in its value correspond to large changes in posterior state distribution, it is worthwhile to investigate whether a similar statement can be made about the burn-in phase using the smoothing estimator,

which, by virtue of being Bayes, is bounded by the minimax risk as well. Specifically, one might be inclined to conclude that we can expect $\|\widehat{\boldsymbol{f}}_{\text{th}} - \widehat{\boldsymbol{f}}_{\mathcal{M}}\|$ to be small, in analogy to $\|\widehat{\boldsymbol{f}}_{\text{th}} - \widehat{\boldsymbol{f}}_{\boldsymbol{\tau}}\|$, but we will argue that this is not the case.

In the usual decision theoretic settings, risk calculations are devices used to rank the quality of estimators under the different prior assumptions a statistician can make, and the minimax risk bounds the risk under the least-favorable prior knowledge she might have about the parameter in question. As such, any risk calculation will not account for any information not included in the prior. Specifically, the integrated risk (Eq. (3.2)) effectively only integrates over the domain of the posterior, either by discounting the frequentist risk (Eq. (3.1)) by a factor of 0 whenever $\mathcal{P}(\boldsymbol{f}) = 0$, or, equivalently, by discounting the loss whenever $\mathcal{P}(\boldsymbol{f} \mid \boldsymbol{y}) = 0$ in the calculation of the posterior expected loss (Eq. (3.4)). Whenever the true $\boldsymbol{f}$ is outside of the posterior domain, the loss it incurs is completely removed from consideration. Risk therefore describes the loss the statistician *expects* subjectively, as opposed to the loss she is expected to *experience* objectively for a true $\boldsymbol{f}$.

In the context of a Bayesian HMM, the risk of the HMM estimator truly corresponds to the prior expectations we have about the latent parameters, as encoded by our hyperparameters $\boldsymbol{\tau}$. The HMM estimator summarizes the posterior distribution of the Bayesian HMM from which we sample after FBG has converged. The smoothing estimator, on the other hand, summarizes the posterior distribution

$$\mathcal{P}(\boldsymbol{q} \mid \boldsymbol{y}, \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi}),$$

from which we sample the state sequence in each FBG iteration, through the position-wise weighted average of state means. Its risk, too, is bounded by the minimax risk; however, this distribution does not encode subjective prior knowledge in the Bayesian sense: any state sequence for which at least one emission mean is not in $\boldsymbol{\theta}$ has probability zero in both the prior and the posterior. It is extremely unlikely for the true state means to be sampled precisely, especially during burn-in, hence the true loss incurred in sampling is not accounted for in most cases. For this reason, it would be a fallacy to expect that $\|\widehat{\boldsymbol{f}}_{\text{th}} - \widehat{\boldsymbol{f}}_{\mathcal{M}}\|$ is small, since the assumptions in a prior $\mathcal{M}$ are violated unless all its state means are those of $\boldsymbol{f}$. Perhaps not surprisingly, the block boundaries in wavelet compression cannot be expected to follow the most likely state transitions

for arbitrary HMM parameters for $\mathcal{M}$. While this could be seen as problematic, we would argue that in fact this might help with convergence behavior of FBG. Wavelet compression changes the distribution $\mathcal{P}(\boldsymbol{q} \mid \boldsymbol{y}, \mathcal{A}, \boldsymbol{\theta}, \boldsymbol{\pi})$ by suppressing state distributions occurring far away from changes in the true posterior marginals of $\mathcal{P}(\boldsymbol{q} \mid \boldsymbol{y}, \boldsymbol{\tau})$ in the Bayesian HMM during the burn-in phase.

4.3 Heteroscedastic HMM

While we show that Haar wavelet regression can be used as an approximation for the expected posterior state means in a σ-HMM, the squared loss itself is of limited concern when regression is used for compression, as it is the preservation of discontinuities in $\boldsymbol{f}$, induced by states with separated means, that prevents over-compression. This is a concern in particular in that wavelet shrinkage using the universal threshold has been found to underfit the data (Antoniadis & Oppenheim 1995; Fan et al. 1993). Furthermore, results concerning estimators are not easily extended to heteroscedastic $\boldsymbol{y}$, as obtained by an HMM in which emission variances are not the same for all states.

4.3.1 Preservation of discontinuities

It is well-established that wavelet regression yields high-amplitude coefficients for wavelets spanning a discontinuity in $\boldsymbol{f}$. For homoscedastic data, breakpoint preservation at coarser resolution levels follows directly from oracle projection, for which universal thresholding is minimax and no better diagonal estimator can be obtained: Assume that, at scale j, all emissions at $t \in L_j$ have emission mean μ_L, and those in R_j have μ_R. If we assume that $\left\langle \boldsymbol{f}, \boldsymbol{\psi}_{j,k} \right\rangle \geq \sigma$ whenever $\boldsymbol{f}$ has a discontinuity at $b^{\pm}_{j,k}$, oracle projection (Eq. (3.8)) would set

$$a_{j,k} = \begin{cases} 1 & \Delta\mu \geq \frac{2}{\sqrt{2^j}}\sigma \\ 0 & \text{else,} \end{cases}$$

creating discontinuities at $b^{\pm}_{j,k}$ in $\widehat{\boldsymbol{f}}$. Means with sufficiently large separation $\Delta\mu$ are therefore unproblematic. However, large noise variance σ^2 can cause an obvious problem. An oracle

concerned with breakpoint preservation, however, would set

$$a_{j,k} = \begin{cases} 1 & \Delta\mu \geq 0 \\ 0 & \text{else.} \end{cases}$$

Asymptotically, the two are equivalent, as

$$\lim_{j\to\infty} \frac{2}{\sqrt{2^j}}\sigma = 0,$$

meaning that even breakpoints between insufficiently well separated states are indeed preserved for sufficiently long state durations. Notice that any introduction of additional discontinuities around that position would only create additional approximate breakpoints, and the number of spurious discontinuities far away from state transitions is limited, since noise coefficients are removed with high probability.

For the heteroscedastic Gaussian case with general state emission variances, the probability of missing a breakpoint can be quantified directly. The inner product of a Haar wavelet with the data is a sum of Gaussian random variables with means corresponding to the latent parameters $\mu_{\boldsymbol{q}[t]}$ for all t in the support interval. It thus follows directly from the additive properties of the Normal distribution that the coefficients are distributed as

$$\left\langle \boldsymbol{y}, \boldsymbol{\psi}_j \right\rangle \sim N\left(\mu_{\boldsymbol{q},j}, \sigma^2_{\boldsymbol{q},j}\right), \quad \text{with}$$

$$\mu_{\boldsymbol{q},j} \coloneqq \left\langle \boldsymbol{\psi}_j, \boldsymbol{q} \right\rangle = \frac{1}{\sqrt{2^j}}\left(\sum_{t\in L_j} \mu_{\boldsymbol{q}[t]} - \sum_{t\in R_j} \mu_{\boldsymbol{q}[t]}\right), \text{ and}$$

$$\sigma^2_{\boldsymbol{q},j} \coloneqq \left\langle \boldsymbol{\psi}_j, \boldsymbol{\epsilon} \right\rangle = \frac{1}{2^j}\sum_{t\in L_j\cup R_j} \sigma^2_{\boldsymbol{q}[t]}.$$

Note how the mean is determined by the signal coefficient alone, and that the variance, determined solely by the noise coefficients, is always the average noise variance across the support, regardless of j.

Due to the symmetry of the normal distribution, let $\mu_L \geq \mu_R$ and $\Delta\mu \coloneqq (\mu_L - \mu_R)$ w.l.o.g., so that $\left\langle \boldsymbol{\psi}_j, \boldsymbol{q} \right\rangle = \frac{\sqrt{2^j}}{2}\Delta\mu$ is non-negative. It follows that

$$\left\langle \boldsymbol{\psi}_j, \boldsymbol{\epsilon} \right\rangle \sim N\left(0, \frac{\sigma_L^2 + \sigma_R^2}{2}\right)$$

and hence

$$\left\langle \boldsymbol{\psi}_j, \boldsymbol{y} \right\rangle = \frac{\sqrt{2^j}}{2}\Delta\mu + \left\langle \boldsymbol{\psi}_j, \boldsymbol{\epsilon} \right\rangle \sim N\left(\frac{\sqrt{2^j}}{2}\Delta\mu, \frac{\sigma_L^2 + \sigma_R^2}{2} \right).$$

For any given threshold λ, we can use this density to directly quantify the probability of under-compressing a state transition as

$$\mathbb{P}\left(\left\langle \boldsymbol{y}, \boldsymbol{\psi}_j \right\rangle \in [-\lambda, \lambda] \right) = \int_{-\lambda}^{\lambda} \frac{1}{\sqrt{\pi(\sigma_L^2 + \sigma_R^2)}} \exp\left(-\frac{\left(\frac{\sqrt{2^j}}{2}\Delta\mu - x \right)^2}{\sigma_L^2 + \sigma_R^2} \right) \mathrm{d}x\,.$$

This integral tends to 0 for a large shift $\Delta\mu$ away from 0, smaller sums of noise variances, or larger scale j. Note that this distribution could also be used directly if a specific mean difference is to be resolved with a given probability at a given scale. In practice, however, we found that such measures were unnecessary for the data we analyzed.

4.3.2 Dynamic Haar compression

The homoscedastic case does not typically occur in practice. Even on experimental platforms such as aCGH, non-diploid states tend to have higher variance. As a result, a direct estimate of emission variances like the one above is not available. Instead, assume $\boldsymbol{\theta}$ was known. As lower thresholds remove less wavelet coefficients, thereby retaining a higher number of discontinuities, which correspond to potential state transitions, using the minimum variance $\sigma^2_{\min} \in \boldsymbol{\theta}$ is a sensible way to create compressed data via wavelet shrinkage. Parts of the data generated by states whose emission variance equals $\sigma^2_{\min}$ will be compressed as before due to polynomial suppression. Higher-variance regions will contain wavelets higher than expected for T observations of minimum variance, thus retaining additional wavelets and discontinuities. As the set of breakpoints for higher thresholds is a subset of those for lower ones, these regions will be compressed like regions with noise variance $\sigma^2_{\min}$, with additional, superfluous block boundaries thrown in. In other words, high-variance regions will be under-compressed, but no block boundary will be missed which wouldn't be missed in the uniform variance case as well.

Unfortunately, $\boldsymbol{\theta}$ is typically latent and has to be inferred. FBG converges to the correct $\boldsymbol{\theta}$ over time, providing increasingly accurate *a priori* samples at each iteration. Therefore, samples of $\boldsymbol{\theta}$ can be used to derive approximate noise levels. We hence propose the following simple approach: In each FBG sampling iteration, we use the smallest sampled variance parameter $\sigma^2_{\min}$

Algorithm 1 Dynamically adaptive FBG for HMMs

procedure HAMMLET($\boldsymbol{y}$, $\tau_{\mathcal{A}}$, τ_θ, τ_π)
  $T \leftarrow |\boldsymbol{y}|$ ▷ get data size
  $\lambda \leftarrow \sqrt{2 \ln T}$ ▷ constant factor in universal threshold
  $\mathcal{A} \sim \mathcal{P}(\mathcal{A} \mid \tau_{\mathcal{A}})$ ▷ prior sampling of transition probabilities
  $\theta \sim \mathcal{P}(\theta \mid \tau_\theta)$ ▷ prior sampling of emission parameters
  $\boldsymbol{\pi} \sim \mathcal{P}(\boldsymbol{\pi} \mid \tau_\pi)$ ▷ prior sampling of state distribution
  for $i \leftarrow 1, \ldots, N$ **do** ▷ iterate Gibbs sampler
    $\sigma_{\min} \leftarrow \min_{\sigma_i} \{\widehat{\sigma}_{\text{MAD}}, \sigma_i \mid \sigma_i^2 \in \theta\}$ ▷ minimum emission variance
    Create block sequence $\boldsymbol{B}$ from threshold $\lambda \sigma_{\min}$
    $\boldsymbol{q} \sim \mathcal{P}(\boldsymbol{q} \mid \mathcal{A}, \boldsymbol{B}, \theta, \boldsymbol{\pi})$ using Forward-Backward sampling
    Add count of marginal states for $\boldsymbol{q}$ to result
    $\mathcal{A} \sim \mathcal{P}\left(\mathcal{A} \,\middle|\, \tau_{\mathcal{A}}^\star\right) = \mathcal{P}(\mathcal{A} \mid \boldsymbol{\pi}, \boldsymbol{q}, \tau_{\mathcal{A}}) \propto \mathcal{P}(\boldsymbol{\pi}, \boldsymbol{q} \mid \mathcal{A}) \mathcal{P}(\mathcal{A} \mid \tau_{\mathcal{A}})$
    $\theta \sim \mathcal{P}\left(\theta \,\middle|\, \tau_\theta^\star\right) = \mathcal{P}(\theta \mid \boldsymbol{q}, \boldsymbol{B}, \tau_\theta) \propto \mathcal{P}(\boldsymbol{q}, \boldsymbol{B} \mid \theta) \mathcal{P}(\theta \mid \tau_\theta)$
    $\boldsymbol{\pi} \sim \mathcal{P}\left(\boldsymbol{\pi} \,\middle|\, \tau_\pi^\star\right) = \mathcal{P}(\boldsymbol{\pi} \mid \mathcal{A}, \boldsymbol{q}, \tau_\pi) \propto \mathcal{P}(\mathcal{A}, \boldsymbol{q} \mid \boldsymbol{\pi}) \mathcal{P}(\boldsymbol{\pi} \mid \tau_\pi)$

to create a new block sequence via wavelet thresholding (Algorithm 1). The method is illustrated in Fig. 4.2.

A potential problem could arise if all sampled variances are too large. In this case, blocks would be under-segmented, yield wrong posterior variances and hide possible state transitions. As a safeguard against over-compression, we use σ^2_{MAD} as an estimate of the variance in the dominant component, and modify the threshold definition to $\lambda \cdot \min\{\widehat{\sigma}_{\text{MAD}}, \sigma_i \in \theta\}$. If the data is not i.i.d., $\widehat{\sigma}^2_{\text{MAD}}$ will systematically underestimate the true variance (WANG & WANG 2007). In this case, the blocks get smaller than necessary, thus decreasing the compression.

4.3.3 The wavelet tree data structure

The necessity to recreate a new block sequence in each iteration based on the most recent estimate of the smallest variance parameter creates the challenge of doing so with little computational overhead, specifically without repeatedly computing the inverse wavelet transform or considering all T elements in other ways. We achieve this by creating a simple tree-based data structure.

The pyramid algorithm yields $\boldsymbol{d}$ sorted according to (j,k). Again, let $h := \operatorname{ld} T$, and $j := h - j$. We can map the wavelet $\psi_{j,k}$ to a perfect binary tree of height h such that all wavelets for scale j are nodes on level j, nodes within each level are sorted according to k, and j is increasing from the leaves to the root (Fig. 4.3). The vector $\boldsymbol{d}$ represents a breadth-first search (BFS) traversal of that tree, with $d_{j,k}$ being the entry at position $\lfloor 2^j \rfloor + k$. Adding $\boldsymbol{y}[i]$ as the i-th leaf on level $j = 0$,

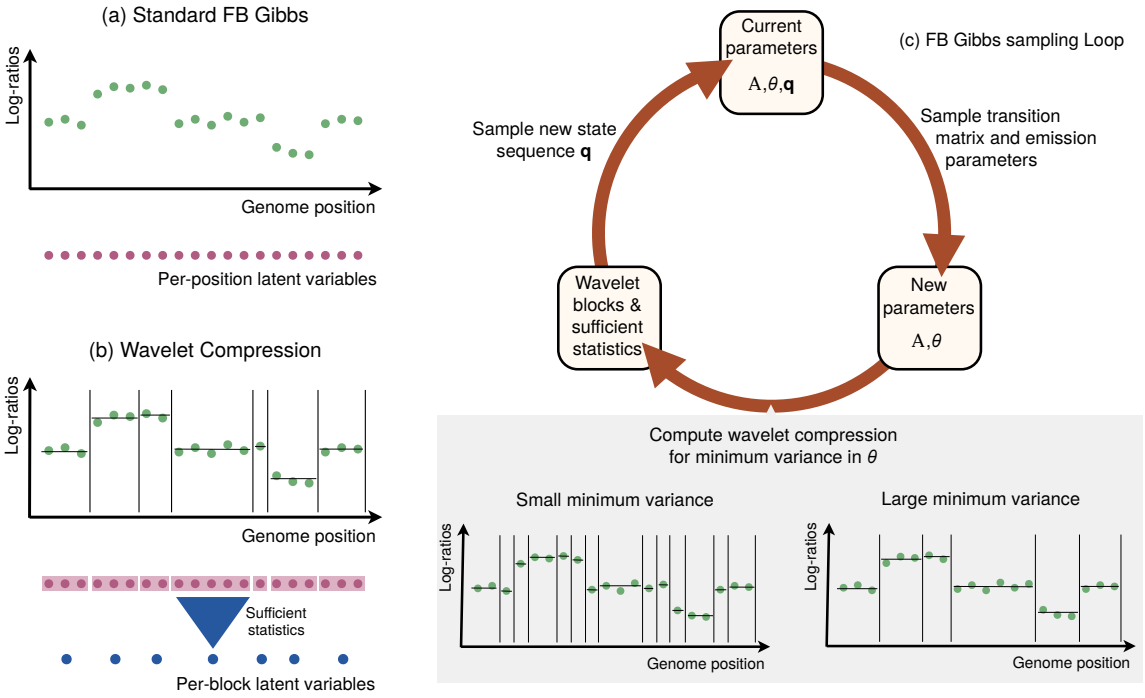


Figure 4.2: Overview of HaMMLET. Instead of individual computations per observation (panel a), Forward-Backward Gibbs Sampling is performed on a compressed version of the data, using sufficient statistics for block-wise computations (panel b) to accelerate inference in Bayesian Hidden Markov Models. During the sampling (panel c) parameters and copy number sequences are sampled iteratively. During each iteration, the sampled variances determine which coefficients of the data's Haar wavelet transform are dynamically set to zero. This controls potential break points at finer or coarser resolution or, equivalently, defines blocks of variable number and size (panel c, bottom).

each non-leaf node represents a wavelet which is non-zero for the $n := 2^j$ data points $\boldsymbol{y}[t]$, for t in the interval $I_{j,k} := [kn, (k+1)n-1]$ stored in the leaves below; notice that for the leaves, $kn = t$.

This implies that the leaves in any subtree all have the same value after wavelet thresholding if all the wavelets in this subtree are set to zero. We can hence avoid computing the inverse wavelet transform to create blocks. Instead, each node stores the maximum absolute wavelet coefficient in the entire subtree, as well as the sufficient statistics required for calculating the likelihood function. More formally, a node $N_{j,t}$ corresponds to wavelet $\psi_{j,k}$, with $j = h - j$ and $t = k2^j$ ($\psi_{-1,0}$ is simply constant on the $[0,1)$ interval and has no effect on block creation, thus we discard it). Essentially, j numbers the levels beginning at the leaves, and t marks the start position of the block when pruning the subtree rooted at $N_{j,t}$. The members stored in each node are:

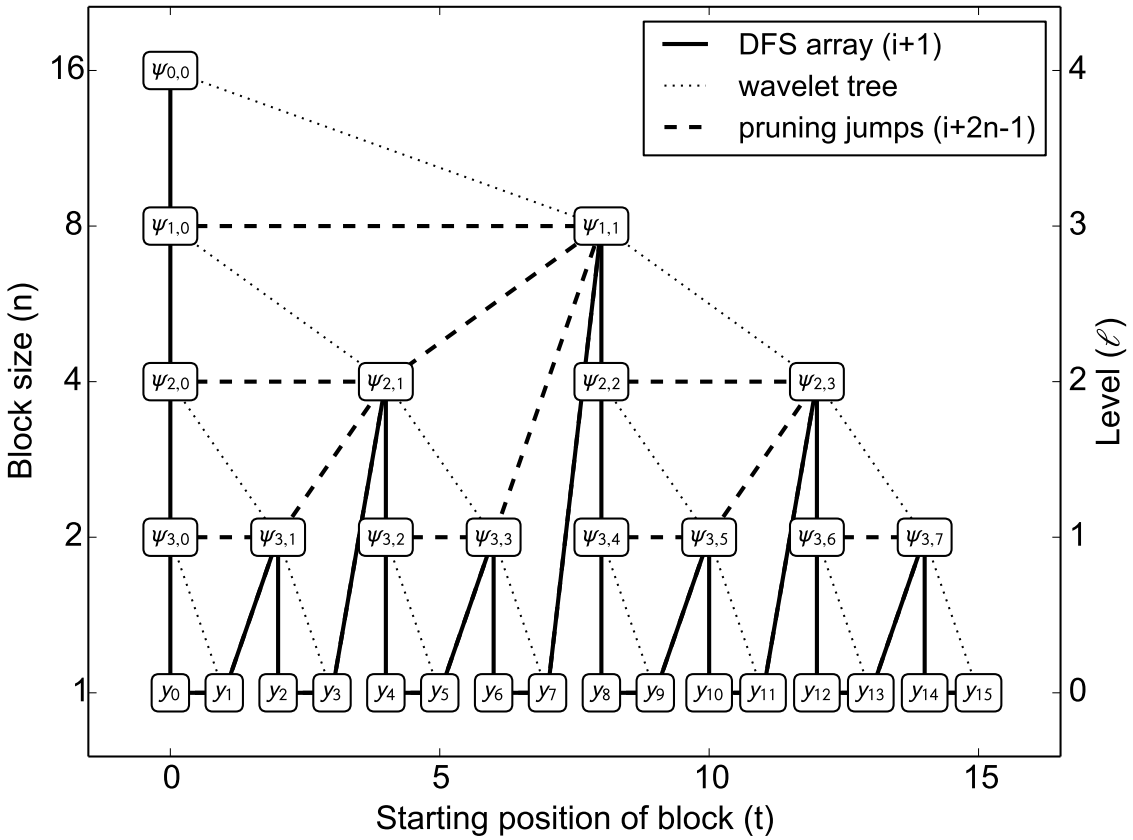


Figure 4.3: Mapping of wavelets $\psi_{j,k}$ and data points y_t to tree nodes $N_{j,t}$. Each node is the root of a subtree with $n = 2^j$ leaves; pruning that subtree yields a block of size n, starting at position t. For instance, the node $N_{1,6}$ is located at position 13 of the DFS array (solid line), and corresponds to the wavelet $\psi_{3,3}$. A block of size $n = 2$ can be created by pruning the subtree, which amounts to advancing by $2n - 1 = 3$ positions (dashed line), yielding $N_{3,8}$ at position 16, which is the wavelet $\psi_{1,1}$. Thus the number of steps for creating blocks per iteration is at most the number of nodes in the tree, and thus strictly smaller than $2T$.

- The number of leaves, corresponding to the block size:

$$N_{j,t}[n] \coloneqq 2^j$$

- The sum of data points stored in the subtree leaves:

$$N_{j,t}[\Sigma_1] \coloneqq \sum_{i \in I_{j,k}} \boldsymbol{y}[i]$$

- Similarly, the sum of squares:

$$N_{j,t}[\Sigma_2] \coloneqq \sum_{i \in I_{j,k}} \boldsymbol{y}[i]^2$$

- The maximum absolute wavelet coefficient of the subtree, including the current $d_{j,k}$ itself:

$$N_{0,t}[d] := 0 \qquad N_{j>0,t}[d] := \max_{\substack{j' \le j \\ t \le t' < t+2^j}} \left\{ \left| d_{h-j',2^{j'}/t'} \right| \right\}$$

All these values can be computed recursively from the child nodes in linear time. As some real data sets contain salt-and-pepper noise, which manifests as isolated large coefficients on the lowest level, its is possible to ignore the first level in the maximum computation so that no information to create a single-element block for outliers is passed up the tree. We refer to this technique as *noise control*. Notice that this does not imply that blocks are only created at even t, since true transitions manifest in coefficients on multiple levels.

The block creation algorithm is simple: upon construction, the tree is converted to *depth-first search* (DFS) order, which simply amounts to sorting the BFS array according to (kn, j), and can be performed using linear-time algorithms such as radix sort; internally, we implemented a different linear-time implementation mimicking tree traversal using a stack. Given a threshold, the tree is then traversed in DFS order by iterating linearly over the array (Fig. 4.3, solid lines). Once the maximum coefficient stored in a node is less or equal to the threshold, a block of size n is created, and the entire subtree is skipped (dashed lines). As the tree is perfect binary and complete, the next array position in DFS traversal after pruning the subtree rooted at the node at index i is simply obtained as $i + 2n - 1$, so no expensive pointer structure needs to be maintained, leaving the tree data structure a simple flat array. An example of dynamic block creation is given in Fig. 4.4.

Proposition 4.3.1. *A wavelet tree produces a partition of the data into K blocks in* $\Theta(2K-1) = O(K)$ *time. Thus, each block is created in expected constant time.*

Proof. This follows trivially from the fact that DFS with pruning of a perfect binary tree is equivalent to a DFS traversal on a full binary tree, where each leaf corresponds to a block. By a standard induction argument, a full binary tree with K leaves has $2K - 1$ nodes, each of which is expanded exactly once during block creation. □

Once the Gibbs sampler converges to a set of variances, the block structure is less likely to change. To avoid recreating the same block structure over and over again, we employ a technique

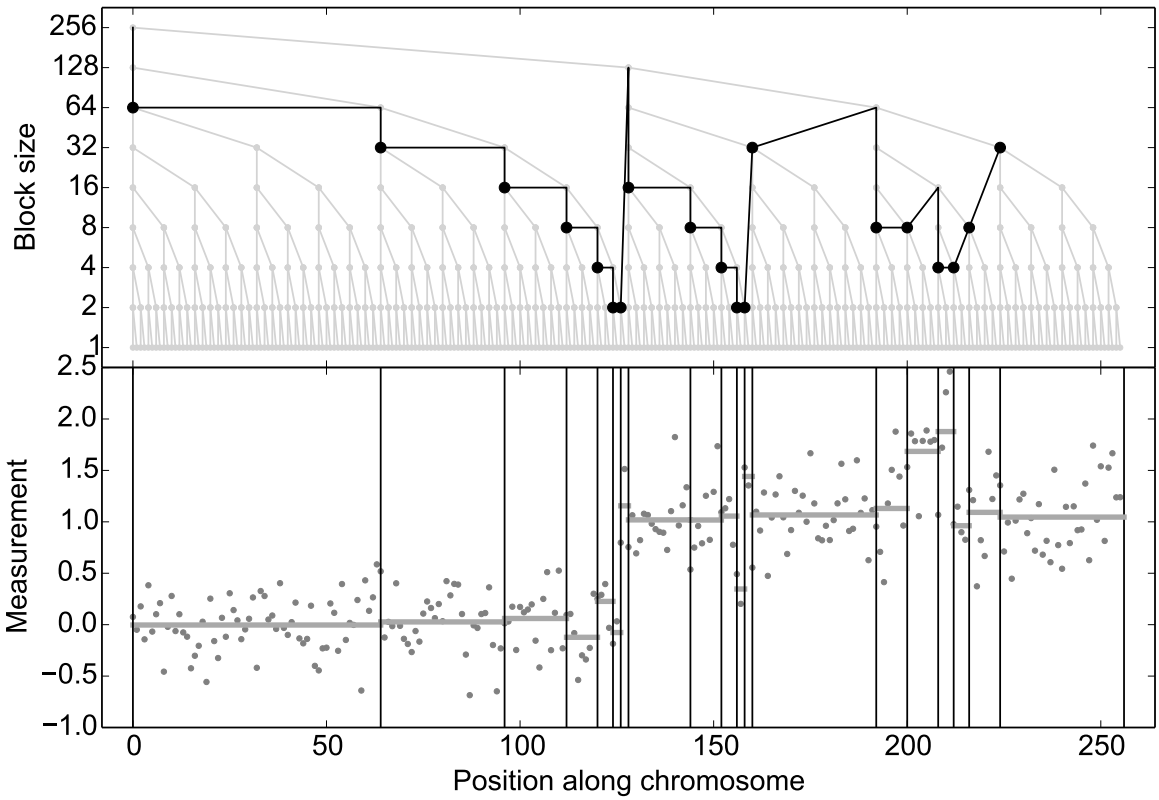


Figure 4.4: Example of dynamic block creation. The data is of size $T = 256$, so the wavelet tree contains 512 nodes. Here, only 37 entries had to be checked against the threshold (dark line), 19 of which (round markers) yielded a block (vertical lines on the bottom). Sampling is hence done on a short array of 19 blocks instead of 256 individual values, thus the compression ratio is 13.5. The horizontal lines in the bottom subplot are the block means derived from the sufficient statistics in the nodes. Notice how the algorithm creates small blocks around the breakpoints, e. g. at $t \approx 125$, which requires traversing to lower levels and thus induces some additional blocks in other parts of the tree (left subtree), since all block sizes are powers of 2. This somewhat reduces the compression ratio, which is unproblematic as it increases the degrees of freedom in the sampler.

called *block structure memoization*. Given a block structure B_1 created by threshold any λ_1, a second, lower threshold $\lambda_2 < \lambda_1$ creates either the same block structure B_1, or a block structure B_2 in which some blocks of B_1 are subdivided into smaller blocks, increasing the number of total blocks. Therefore, we can partition the set of possible values for λ into intervals $[\lambda_i, \lambda_j]$, each of which is associated with a particular, unique number of blocks. Thus, for each block sequence length we register the minimum and maximum variance that creates that sequence. Upon entering a new iteration, we check if the current variance would create the same number of blocks as in the previous iteration, which guarantees that we would obtain the same block sequence, and hence can avoid recomputation.

The wavelet tree data structure can be readily extended to multivariate data of dimensionality

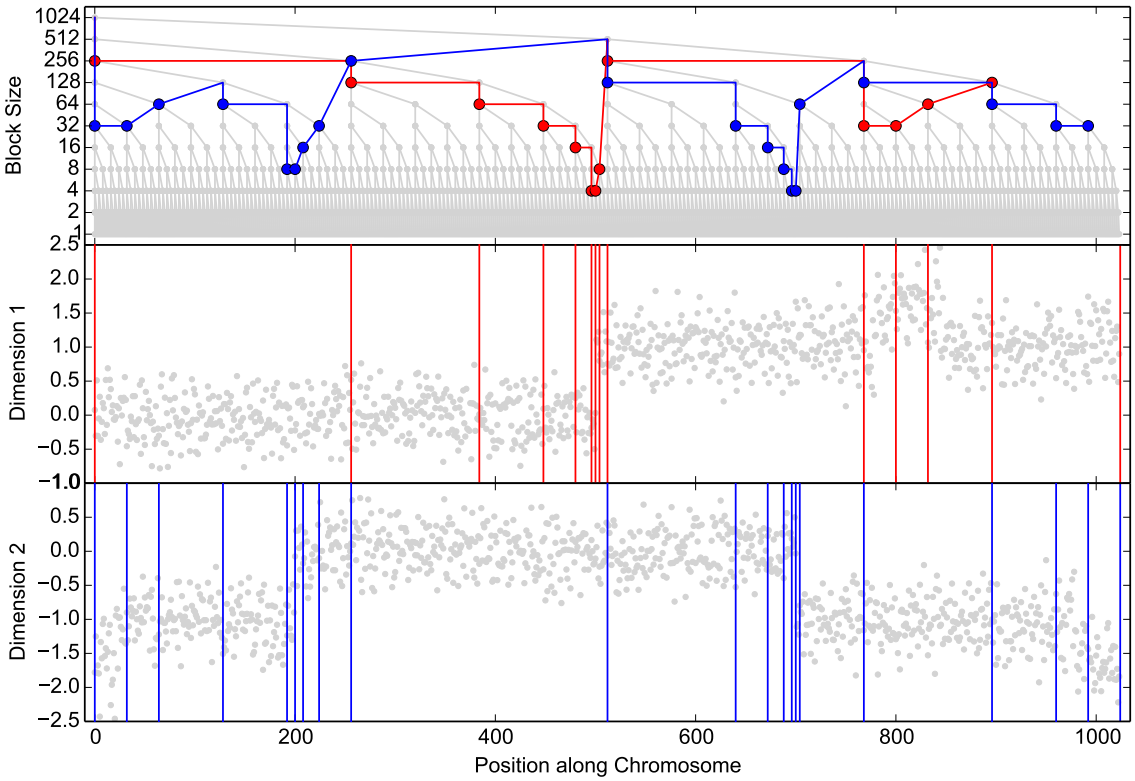


Figure 4.5: An example of a multivariate wavelet tree. The top figure shows the topology of a wavelet tree for each of the two data dimensions below. The red and blue paths denote the pruning paths for the first and second data dimension, respectively. The bottom two subplots show the block boundaries for each dimension. The joint block boundaries for this bivariate data would consist of the union of those boundaries. Since they are incurred by entries above the threshold in their respective wavelet tree, the trees can be merged into one by taking the maximum absolute detail coefficient across dimensions to create the union of block boundaries. Hence, the runtime for finding block boundaries does not depend on the dimensionality of the data.

m (Fig. 4.5). Instead of storing m different trees and reconciling m different block patterns in each iteration, one simply stores m different values for each sufficient statistic in a tree node. Since we have to traverse into the combined tree if the coefficient of any of the m trees was below the threshold, we simply store the largest $N_{j,t}[d]$ among the corresponding nodes of the trees, which means that the block creation can be done in $O(T)$ instead of $O(mT)$, i. e. the dimensionality of the data only enters into the creation of the data structure, but not the query during sampling iterations.

4.3.4 Automatic priors

While Bayesian methods allow for inductive bias such as the expected location of means, it is desirable to be able to use our method even when little domain knowledge exists, or large variation is expected, such as the lab and batch effects commonly observed in micro-arrays (Luo et al. 2010), as well as unknown means due to sample contamination. Since FBG does require a prior even in that case, we propose the following method to specify hyperparameters of a weak prior automatically. Posterior samples of means and variances are drawn from a Normal-Inverse Gamma distribution $(\mu, \sigma^2) \sim \mathrm{NI}\Gamma(\mu_0, \nu, \alpha, \beta)$, whose marginals simply separate into a Normal and an Inverse Gamma distribution

$$\sigma^2 \sim \mathrm{I}\Gamma(\alpha, \beta), \quad \mu \sim N\left(\mu_0, \frac{\sigma^2}{\nu}\right).$$

Let s^2 be a user-defined variance (or automatically inferred, e. g. from the largest of the finest detail coefficients, or $\widehat{\sigma}^2_{\mathrm{MAD}}$), and p the desired probability to sample a variance not larger than s^2. From the CDF of $\mathrm{I}\Gamma$ we obtain

$$p \coloneqq \mathbb{P}(\sigma^2 \leq s^2) = \frac{\Gamma\left(\alpha, \frac{\beta}{s^2}\right)}{\Gamma(\alpha)} = Q\left(\alpha, \frac{\beta}{s^2}\right).$$

$\mathrm{I}\Gamma$ has a mean for $\alpha > 1$, and closed-form solutions for $\alpha \in \mathbb{N}$. Furthermore, $\mathrm{I}\Gamma$ has positive skewness for $\alpha > 3$. We thus let $\alpha = 2$, which yields

$$\beta = -s^2\left(W_{-1}\left(-\frac{p}{e}\right) + 1\right), \quad 0 < p \leq 1,$$

where W_{-1} is the negative branch of the Lambert W-function, which is transcendental. However, an excellent analytical approximation with a maximum error of 0.025% is given in Barry et al. (2000), which yields

$$\beta \approx s^2\left(\frac{2\sqrt{b}}{M_1\sqrt{b} + \sqrt{2}\left(M_2 b \exp\left(M_3\sqrt{b}\right) + 1\right)} + b\right),$$

$$b \coloneqq -\ln p,$$

$$M_1 \coloneqq 0.3361, \quad M_2 \coloneqq -0.0042, \quad M_3 \coloneqq -0.0201.$$

Since the mean of $\mathrm{I}\Gamma$ is $\frac{\beta}{\alpha-1}$, the expected variance of μ is $\frac{\beta}{\nu}$ for $\alpha = 2$. To ensure proper mixing, we could simply set $\frac{\beta}{\mu}$ to the sample variance of the data, which can be estimated from the

sufficient statistics in the root of the wavelet tree (the first entry in the array), provided that $\boldsymbol{\mu}$ contained all states in almost equal number. However, due to possible class imbalance, means for short segments far away from μ_0 can have low sampling probability, as they do not contribute much to the sample variance of the data. We thus define δ to be the sample variance of block means in the compression obtained by $\widehat{\sigma}^2_{\text{MAD}}$, and take the maximum of those two variances. We thus obtain

$$\mu_0 \coloneqq \frac{\Sigma_1}{n}, \text{ and } \nu = \beta \max\left\{\frac{n\Sigma_2 - \Sigma_1^2}{n^2}, \delta\right\}^{-1}.$$

4.3.5 Numerical issues

To assure numerical stability when working with probabilities, many HMM implementations resort to log-space computations, which involves a considerable number of expensive function calls (exp, log, pow); for instance, on Intel's Nehalem architecture, log (FYL2X) requires 55 operations as opposed to 1 for adding and multiplying floating point numbers (FADD, FMUL) (Fog 2016). Our implementation, which differs from (Mahmud & Schliep 2011) greatly reduces the number of such calls by utilizing the block structure: The term accounting for emissions and self-transitions within the block can be written as

$$\frac{A_{jj}^{n_w-1}}{(2\pi)^{n_w/2}\sigma_j^{n_w}} \exp\left(-\sum_{k=1}^{n_w}\frac{(\boldsymbol{y}[w][k]-\mu_j)^2}{2\sigma_j^2}\right).$$

Any constant cancels out during normalization. Furthermore, exponentiation of potentially small numbers causes underflows. We hence move those terms into the exponent, utilizing the much stabler logarithm function.

$$\exp\left(-\sum_{k=1}^{n_w}\frac{(\boldsymbol{y}[w][k]-\mu_j)^2}{2\sigma_j^2} + (n_w-1)\log A_{jj} - n_w \log\sigma_j\right).$$

Using the block's sufficient statistics

$$n_w, \quad \Sigma_1 \coloneqq \sum_{k=1}^{n_w}\boldsymbol{y}[w][k], \quad \Sigma_2 \coloneqq \sum_{k=1}^{n_w}\boldsymbol{y}[w][k]^2$$

the exponent can be rewritten as

$$E_w(j) \coloneqq \frac{2\mu_j\Sigma_1 - \Sigma_2}{2\sigma_j^2} + K(n_w, j),$$

$$K(n_w, j) \coloneqq (n_w-1)\log A_{jj} - n_w\left(\log\sigma_j + \frac{\mu_j^2}{2\sigma_j^2}\right).$$

$K(n_w, j)$ can be precomputed for each iteration, thus greatly reducing the number of expensive function calls. Notice that the expressions above correspond to the canonical exponential family form $\exp(\langle t(x), \theta\rangle - F(\theta) + k(x))$ of a product of Gaussian distributions. Hence, equivalent terms can easily be derived for non-Gaussian emissions, implying that the same optimizations can be used in the general case of exponential family distributions: Only the dot product of the sufficient statistics $t(x)$ and the parameters θ has to be computed in each iteration and for each block, while the log-normalizer $F(\theta)$ can be precomputed for each iteration, and the carrier measure $k(x)$ (which is 0 for Gaussian emissions) only has to be computed once.

To avoid overflow of the exponential function, we subtract the largest such exponents among all states, hence $E_w(j) \leq 0$. This is equivalent to dividing the forward variables by

$$\exp\Big(\max_k E_w(k)\Big),$$

which cancels out during normalization. Hence we obtain

$$\tilde{\alpha}_w(j) := \exp\Big(E_w(j) - \max_k E_w(k)\Big) \sum_{i=1}^{n_w} \alpha_{w-1}(i) A_{ij},$$

which are then normalized to

$$\widehat{\alpha}_w(j) = \frac{\tilde{\alpha}_w(j)}{\sum_k \tilde{\alpha}_w(k)}.$$

4.4 Evaluation

4.4.1 Simulated aCGH data

A previous survey (Lai et al. 2005) of eleven CNV calling methods for aCGH has established that segmentation-focused methods such as DNAcopy (Olshen & Venkatraman 2002; Olshen, Venkatraman, et al. 2004), an implementation of circular binary segmentation (CBS), as well as CGHseg (Picard et al. 2005) perform consistently well. DNAcopy performs a number of t-tests to detect break-point candidates. The result is typically over-segmented and requires a merging step in post-processing, especially to reduce the number of segment means. To this end MergeLevels was introduced by Willenbrock & Fridlyand (2005). They compare the combination DNAcopy+MergeLevels to their own HMM implementation (Fridlyand et al. 2004) as well as GLAD (Hupé et al. 2004), showing its superior performance over both methods.

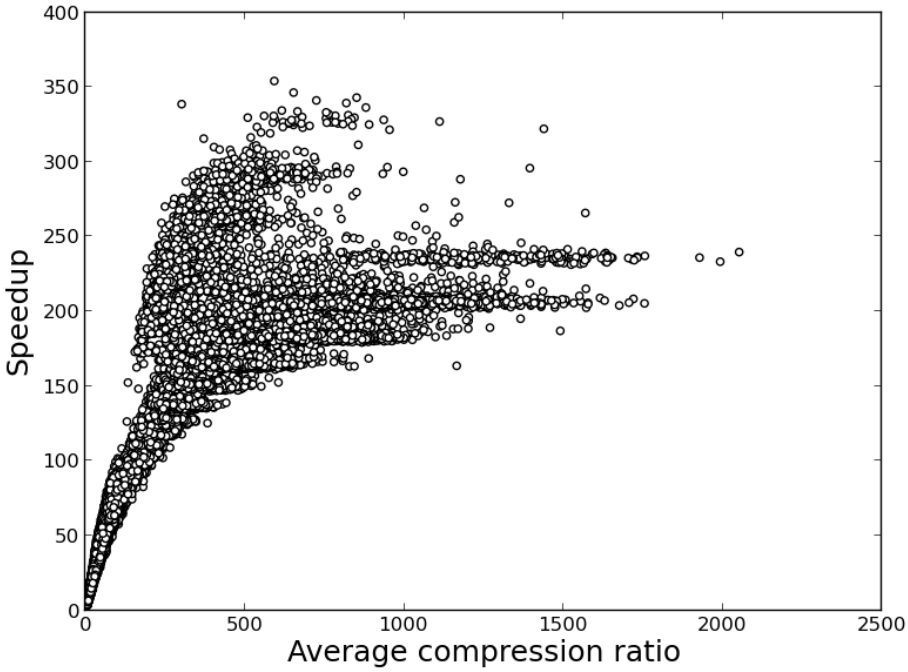


Figure 4.6: HaMMLET's speedup as a function of the average compression during sampling. As expected, higher compression leads to greater speedup. The non-linear characteristic is due to the fact that some overhead is incurred by the dynamic compression, as well as parts of the implementation that do not depend on the compression, such as tallying marginal counts.

This established DNAcopy+MergeLevels as the *de facto* standard in CNV detection, despite the comparatively long running time.

The paper also includes aCGH simulations deemed to be reasonably realistic by the community. DNACopy was used to segment 145 unpublished samples of breast cancer data, and subsequently labeled as copy numbers 0 to 5 by sorting them into bins with boundaries

$$(-\infty, -0.4, -0.2, 0.2, 0.4, 0.6, \infty),$$

based on the sample mean in each segment (the last bin appears to not be used). Empirical length distributions were derived, from which the sizes of CN aberrations are drawn. The data itself is modeled to include Gaussian noise, which has been established as sufficient for aCGH data (Hodgson et al. 2001). Means were generated such as to mimic random tumor cell proportions, and random variances were chosen to simulate experimenter bias often observed in real data; this emphasizes the importance of having automatic priors available when using Bayesian methods, as the means and variances might be unknown *a priori*. The data comprises three sets of simulations: "breakpoint detection and merging" (BD&M), "spatial resolution study" (SRS), and "testing" (T)

(see their paper for details). We used the MergeLevels implementation as provided on their website. It should be noted that the superiority of DNAcopy+MergeLevels was established using a simulation based upon segmentation results of DNAcopy itself.

We used the Bioconductor package DNAcopy (version 1.24.0), and followed the procedure suggested therein, including outlier smoothing. This version uses the linear-time variety of CBS (Venkatraman & Olshen 2007); note that other authors such as Tsourakakis et al. (2011) compare against a quadratic-time version of CBS (Olshen, Venkatraman, et al. 2004), which is significantly slower. For HaMMLET, we use a 5-state model with automatic hyperparameters $\mathbb{P}(\sigma^2 \leq 0.01) = 0.9$ (see Section 4.3.4), and all Dirichlet hyperparameters set to 1.

Following Mahmud & Schliep (2011), we report F-measures (F_1 scores) for binary classification into normal and aberrant segments (Fig. 4.10), using the usual definition of $F = \frac{2\pi\rho}{\pi+\rho}$ being the harmonic mean of precision $\pi = \frac{\mathrm{TP}}{\mathrm{TP+FP}}$ and recall $\rho = \frac{\mathrm{TP}}{\mathrm{TP+FN}}$, where TP, FP, TN and FN denote true/false positives/negatives, respectively. On datasets T and BD&M, both methods have similar medians, but HaMMLET has a much better interquartile range (IQR) and range, about half of CBS's. On the spatial resolution data set (SRS), HaMMLET performs much better on very small aberrations. This might seem somewhat surprising, as short segments could easily get lost under compression. However, Lai et al. (2005) have noted that smoothing-based methods such as quantile smoothing (quantreg) (Eilers & de Menezes 2005), lowess (Cleveland 1979), and wavelet smoothing (Hsu et al. 2005) perform particularly well in the presence of high noise and small CN aberrations, suggesting that "an optimal combination of the smoothing step and the segmentation step may result in improved performance". Our wavelet-based compression inherits those properties. For CNVs of sizes between 5 and 10, CBS and HaMMLET have similar ranges, with CBS being skewed towards better values; CBS has a slightly higher median for 10–20, with IQR and range being about the same. However, while HaMMLET's F-measure consistently approaches 1 for larger aberrations, CBS does not appear to significantly improve after size 10. The plots for all individual samples can be found in Web Supplement S1–S3, which can be viewed online at http://schlieplab.org/Supplements/HaMMLET/, or downloaded from https://zenodo.org/record/46263 (DOI: 10.5281/zenodo.46263).

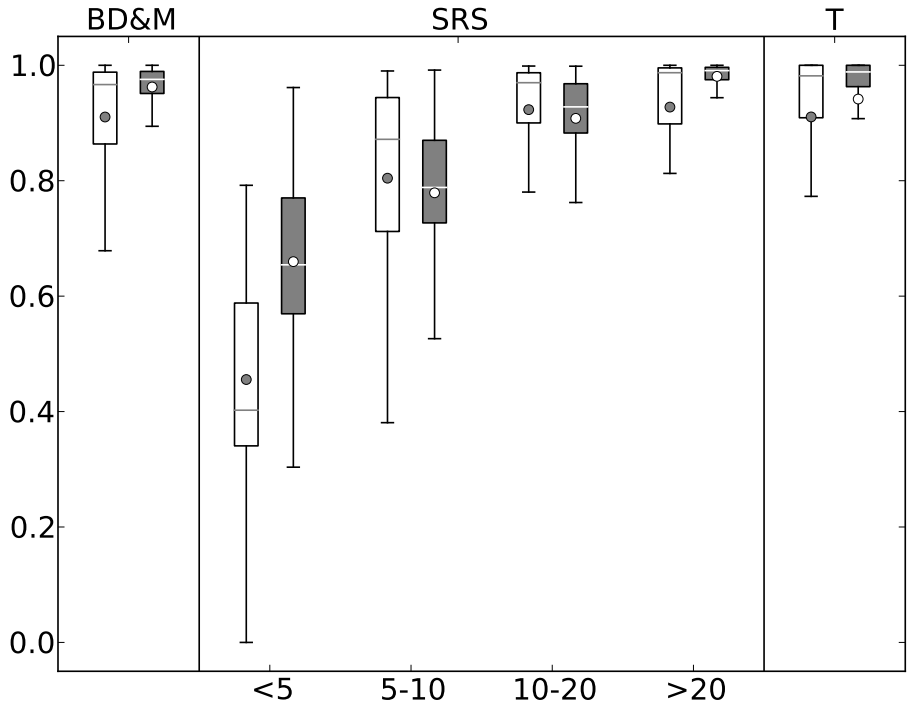


Figure 4.7: F-measures of CBS (light) and HaMMLET (dark) for calling aberrant copy numbers on simulated aCGH data (Willenbrock & Fridlyand 2005). Boxes represent the interquartile range (IQR = Q3−Q1), with a horizontal line showing the median (Q2), whiskers representing the range ($\frac{3}{2}$IQR beyond Q1 and Q3), and the bullet representing the mean. HaMMLET has the same or better F-measures in most cases, and on the SRS simulation converges to 1 for larger segments, whereas CBS plateaus for aberrations greater than 10.

4.4.2 High-density CGH array

In this section, we demonstrate HaMMLET's performance on biological data. Due to the lack of a gold standard for high-resolution platforms, we assess the CNV calls qualitatively. We use raw aCGH data (GEO:GSE23949) (Edgren et al. 2011) of genomic DNA from breast cancer cell line BT-474 (invasive ductal carcinoma, GEO:GSM590105), on an Agilent-021529 Human CGH Whole Genome Microarray 1x1M platform (GEO:GPL8736). We excluded gonosomes, mitochondrial and random chromosomes from the data, leaving 966,432 probes in total.

HaMMLET allows for using automatic emission priors (see Section 4.3.4) by specifying a noise variance, and a probability to sample a variance not exceeding this value. We compare HaMMLET's performance against CBS, using a 20-state model with automatic priors, $\mathbb{P}\left(\sigma^2 \leq 0.1\right) = 0.8$, 10 prior self-transitions and 1 for all other hyperparameters. CBS took over 2 h 9 min to process the entire array, whereas HaMMLET took 27.1 s for 100 iterations, a speedup of 288. The compression

ratio (see Section 4.4.3) was 220.3. CBS yielded a massive over-segmentation into 1,548 different copy number levels; cf. Supplement S4 at https://zenodo.org/record/46263. As the data is derived from a relatively homogeneous cell line as opposed to a biopsy, we do not expect the presence of subclonal populations to be a contributing factor (Burdall et al. 2003; Holliday & Speirs 2011). Instead, measurements on aCGH are known to be spatially correlated, resulting in a wave pattern which has to be removed in a preprocessing step; notice that the internal compression mechanism of HaMMLET is derived from a spatially adaptive regression method, so smoothing is inherent to our method. CBS performs such a smoothing, yet an unrealistically large number of different levels remains, likely due to residuals of said wave pattern. Furthermore, repeated runs of CBS yielded different numbers of levels, suggesting that indeed the merging was incomplete. This can cause considerable problems downstream, as many methods operate on labeled data. A common approach is to consider a small number of classes, typically 3 to 4, and associate them semantically with CN labels like *loss*, *neutral*, *gain*, and *amplification*, e.g. van Wieringen, van ve Wiel & Ylstra (2008), Liu et al. (2006), González et al. (2009), van de Wiel & van Wieringen (2007), Shah, Lam, et al. (2007), Guha, Li & Neuberg (2006), Yin & Li (2009), Hodgson et al. (2001), and Hupé et al. (2004). In inference models that contain latent categorical state variables, like HMM, such an association is readily achieved by sorting classes according to their means. In contrast, methods like CBS typically yield a large, often unbounded number of classes, and reducing it is the declared purpose of merging algorithms, see Willenbrock & Fridlyand (2005). Consider, for instance, CGHregions (van de Wiel & van Wieringen 2007), which uses a 3-label matrix to define regions of shared CNV events across multiple samples by requiring a maximum L_1 distance of label signatures between all probes in that region. If the domain of class labels was unrestricted and potentially different in size for each sample, such a measure would not be meaningful, since the i-th out of n classes cannot be readily identified with the i-th out of m classes for $n \neq m$, hence no two classes can be said to represent the same CN label. Similar arguments hold true for clustering based on Hamming distance (Liu et al. 2006) or ordinal similarity measures (van Wieringen, van ve Wiel & Ylstra 2008). Furthermore, even CGHregions' optimized computation of medoids takes several minutes to compute. As the time depends multiplicatively on the number of labels, increasing it by three orders of magnitude would increase downstream running times to many hours.

For a more comprehensive analysis, we restricted our evaluation to chromosome 20 (21,687 probes), which we assessed to be the most complicated to infer, as it appears to have the highest number of different CN states and breakpoints. CBS yields a 19-state result after 15.78 s (Fig. 4.8, top). We have then used a 19-state model with automated priors ($\mathbb{P}\left(\sigma^2 \leq 0.04\right) = 0.9$), 10 prior self-transitions, all other Dirichlet parameters set to 1) to reproduce this result. Using noise control (see Section 4.3.3), our method took 1.61 s for 600 iterations. The solution we obtained is consistent with CBS (Fig. 4.8, middle and bottom). However, only 11 states were part of the final solution, i. e. 8 states yielded no significant likelihood above that of other states. We observe superfluous states being ignored in our simulations as well. In light of the results on the entire array, we suggest that the segmentation by DNAcopy has not sufficiently been merged by MergeLevels. Most strikingly, HaMMLET does not show any marginal support for a segment called by CBS around probe number 4,500. We have confirmed that this is not due to data compression, as the segment is broken up into multiple blocks in each iteration (cf. Supplement S5 at https://zenodo.org/record/46263). On the other hand, two much smaller segments called by CBS in the 17,000–20,000 range do have marginal support of about 40% in HaMMLET, suggesting that the lack of support for the larger segment is correct. It should be noted that inference differs between the entire array and chromosome 20 in both methods, since long-range effects have higher impact in larger data.

We also demonstrate another feature of HaMMLET called *noise control*. While Gaussian emissions have been deemed a sufficiently accurate noise model for aCGH (Hodgson et al. 2001), microarray data is prone to outliers, for example due to damages on the chip. While it is possible to model outliers directly (Shah, Xuan, et al. 2006), the characteristics of the wavelet transform allow us to largely suppress them during the construction of our data structure. Notice that due to noise control most outliers are correctly labeled according to the segment they occur in, while the short gain segment close to the beginning is called correctly.

4.4.3 Effects of wavelet compression on speed and convergence

The speedup gained by compression depends on how well the data can be compressed. Poor compression is expected when the means are not well separated, or short segments have small variance, which necessitates the creation of smaller blocks for the rest of the data to expose

potential low-variance segments to the sampler. On the other hand, data must not be over-compressed to avoid merging small aberrations with normal segments, which would decrease the F-measure. Due to the dynamic changes to the block structure, we measure the level of compression as the average compression ratio, defined as the product of the number of data points T and the number of iterations N, divided by the total number of blocks in all iterations. As usual a compression ratio of one indicates no compression.

To evaluate the impact of dynamic wavelet compression on speed and convergence properties of an HMM, we created 129,600 different data sets with $T = 32{,}768$ many probes. In each data set, we randomly distributed 1 to 6 gains of a total length of $\{100, 250, 500, 750, 1000\}$ uniformly among the data, and do the same for losses. Mean combinations

$$(\mu_{\text{loss}}, \mu_{\text{neutral}}, \mu_{\text{gain}})$$

were chosen from $(\operatorname{ld}\frac{1}{2}, \operatorname{ld}1, \operatorname{ld}\frac{3}{2})$, $(-1, 0, 1)$, $(-2, 0, 2)$, and $(-10,\ 0,\ 10)$, and variances

$$(\sigma^2_{\text{loss}}, \sigma^2_{\text{neutral}}, \sigma^2_{\text{gain}})$$

where chosen from $(0.05, 0.05, 0.05)$, $(0.5, 0.1, 0.9)$, $(0.3, 0.2, 0.1)$, $(0.2, 0.1, 0.3)$, $(0.1, 0.3, 0.2)$, and $(0.1, 0.1, 0.1)$. These values have been selected to yield a wide range of easy and hard cases, both well separated, low-variance data with large aberrant segments as well as cases in which small aberrations overlap significantly with the tail samples of high-variance neutral segments; an example of a hard inference task is shown in Fig. 4.9. Consequently, compression ratios range from ∼1 to ∼2,100. We use automatic priors $\mathbb{P}(\sigma^2 \leq 0.2) = 0.9$, self-transition priors $\alpha_{ii} \in \{10, 100, 1000\}$, non-self transition priors $\alpha_{ij} = 1$, and initial state priors $\boldsymbol{\alpha} \in \{\mathbf{1}, \mathbf{10}\}$. Using all possible combinations of the above yields 129,600 different simulated data sets, a total of 4.2 billion values.

We achieve speedups per iteration of up to 350 compared to an uncompressed HMM (Fig. 4.6). In contrast, Mahmud & Schliep (2011) have reported ratios of 10–60, with one instance of 90. Notice that the speedup is not linear in the compression ratio. While sampling itself is expected to yield linear speedup, the marginal counts still have to be tallied individually for each position, and dynamic block creation causes some overhead. The quantization artifacts observed for larger speedup are likely due to the limited resolution of the Linux time command (10 ms). Compressed

HaMMLET took about 11.2 CPU hours for all 129,600 simulations, whereas the uncompressed version took over 3 weeks and 5 days. All running times reported are CPU time measured on a single core of a AMD Opteron 6174 Processor, clocked at 2.2 GHz.

We evaluate the convergence of the F-measure of compressed and uncompressed inference for each simulation. Since we are dealing with multi-class classification, we use the micro- and macro-averaged F-measures (F_{mi}, F_{ma}) proposed by Özgür, Özgür & Güngör (2005):

$$F_{\mathrm{mi}} = \frac{2\pi\rho}{\pi+\rho} \quad \text{with} \quad \pi = \frac{\sum_{i=1}^{M} \mathrm{TP}_i}{\sum_{i=1}^{M}(\mathrm{TP}_i + \mathrm{FP}_i)}, \quad \rho = \frac{\sum_{i=1}^{M} \mathrm{TP}_i}{\sum_{i=1}^{M}(\mathrm{TP}_i + \mathrm{FN}_i)} \quad \text{and}$$

$$F_{\mathrm{ma}} = \frac{\sum_{i=1}^{M} F_i}{M} \quad \text{with} \quad \pi_i = \frac{\mathrm{TP}_i}{\mathrm{TP}_i + \mathrm{FP}_i}, \quad \rho_i = \frac{\mathrm{TP}_i}{\mathrm{TP}_i + \mathrm{FN}_i}, \quad F_i = \frac{2\pi_i\rho_i}{\pi_i + \rho_i}.$$

Here, TP_i denotes a true positive call for the i-th out of M states, π and ρ denote precision and recall. These F-measures tend to be dominated by the classifier's performance on common and rare categories, respectively. Since all state labels are sampled from the same prior and hence their relative order is random, we used the label permutation which yielded the highest sum of micro- and macro-averaged F-measures. Instead of finding this permutation by brute force, which requires $K!$ computations for k states, we solve a combinatorial *assignment problem*: Let $\mathcal{C}$ be the confusion matrix of the segmentation, i. e. $\mathcal{C}_{tp}$ is the number of true observations of state t which have been predicted as state p. Notice that $\mathrm{TP}_i + \mathrm{FP}_i$ is just the sum C_i of the i-th column of $\mathcal{C}$, and likewise $\mathrm{TP}_i + \mathrm{FN}_i$ is the sum R_i of the i-th row. Hence, F_{mi} can be computed as $\frac{\operatorname{tr}\mathcal{C}}{\sum_{tp}\mathcal{C}_{tp}}$. The best F_{mi} is thus obtained by the column permutation which maximizes the trace. Likewise, let

$$\pi_{tp} \coloneqq \frac{\mathcal{C}_{tp}}{C_p}, \quad \rho_{tp} \coloneqq \frac{\mathcal{C}_{tp}}{R_t}, \quad F_{tp} \coloneqq \frac{2\pi_{tp}\rho_{tp}}{\pi_{tp}+\rho_{tp}}.$$

Then

$$F_{\mathrm{mi}} + F_{\mathrm{ma}} = \operatorname{tr}\mathcal{M}$$

for

$$\mathcal{M}_{tp} \coloneqq \frac{2\pi_{tp}\rho_{tp}}{K(\pi_{tp}+\rho_{tp})} + \frac{\mathcal{C}_{tp}}{\sum_{i,j}\mathcal{C}_{tp}},$$

and the maximum sum is attained by the column permutation which maximizes the trace. This is the *maximum weighted bipartite matching problem*, which can easily be transformed into the assignment problem and solved in $O(K^3)$ time using the *Hungarian method* due to Kőnig,

Egerváry and Munkres (Kuhn 1955). The simulation results are included in Supplement S6 at https://zenodo.org/record/46263.

In Fig. 4.10, we show that the compressed version of the Gibbs sampler converges almost instantly, whereas the uncompressed version converges much slower, with about 5% of the cases failing to yield an F-measure > 0.6 within 1,000 iterations. Wavelet compression is likely to yield reasonably large blocks for the majority class early on, which leads to a strong posterior estimate of its parameters and self-transition probabilities. As expected, F_{ma} are generally worse, since any misclassification in a rare class has a larger impact. Especially in the uncompressed version, we observe that F_{ma} tends to plateau until F_{mi} approaches 1.0. Since any misclassification in the majority (neutral) class adds false positives to the minority classes, this effect is expected. It implies that correct labeling of the majority class is a necessary condition for correct labeling of minority classes, in other words, correct identification of the rare, interesting segments requires the sampler to properly converge, which is much harder to achieve without compression. It should be noted that running compressed HaMMLET for 1,000 iterations is unnecessary on the simulated data, as in all cases it converges between 25 and 50 iterations. Thus, for all practical purposes, further speedup by a factor of 40–80 can be achieved by reducing the number of iterations, which yields convergence up to 3 orders of magnitude faster than standard FBG.

4.4.4 Coriell, ATCC and breast carcinoma

The data provided by Snijders, Nowak, et al. (2001) includes 15 aCGH samples for the Coriell cell line. At about 2,000 probes, the data is small compared to modern high-density arrays. Nevertheless, these data sets have become a common standard to evaluate CNV calling methods, as they contain few and simple aberrations. The data also contains 6 ATCC cell lines as well as 4 breast carcinoma, all of which are substantially more complicated, and typically not used in software evaluations. In Fig. 4.11, we demonstrate our ability to infer the correct segments on the most complex example, a T47D breast ductal carcinoma sample of a 54 year old female. We used 6-state automatic priors with $\mathbb{P}\left(\sigma^2 \leq 0.1\right) = 0.85$, and all Dirichlet hyperparameters set to 1. On a standard laptop, HaMMLET took 0.09 seconds for 1,000 iterations; running times for the other samples were similar. Our results for all 25 data sets have been included in Supplement S7 at https://zenodo.org/record/46263.

4.5 Forward bias and convergence behavior

We have shown that, compared to uncompressed FBG, state sequences sampled from HaMMLET very quickly approach the ground truth. If initial iterations are treated as burn-in, tallying the maximum marginals approaches the MPM segmentation. Since this behavior is measured in the number of iterations, not CPU time, the qualitative differences must be due to features of wavelet compression other than the mere reduction in data size.

Compressing the data into blocks assumes that all emissions within the block were generated by the same latent HMM state. However, this ignores the contribution of state paths that switch states within the block, and raise the question of bias in the sampler. A typical assumption is that the overall contribution of such paths is small, since non-generating states will yield low emission likelihoods for the observed data, and off-diagonal probabilities in $\mathcal{A}$ are often small in practice. Mahmud & Schliep (2011) showed that this *weak path assumption* holds for a homoscedastic Gaussian HMM with well-separated means if all emissions within a block are closer to the mean of their generating state than to those of any other state. The proof does not distinguish between the parameters of the generating HMM and the ones used in the forward recursion, meaning that, for the proof to hold, we have to assume strong emission posteriors right from the beginning. Of course, a Gibbs sampler might not necessarily yield some μ_x for which these assumptions hold, especially during the burn-in phase. We therefore derive the multiplicative error of normalized forward variables under arbitrary compression and for general—not necessarily EFD, homoscedastic or univariate—HMM:

Definition 4.5.1 (Forward bias)**.** *Let H be a K-state Hidden Markov Model states, such that all emission likelihoods are positive. Let $\boldsymbol{\alpha}_t[s]$ be the normalized forward variable of state s at position t. Let $\widehat{\boldsymbol{\alpha}}_t[s]$ be the normalized approximate forward variable calculated from $\boldsymbol{\alpha}_t$ by ignoring transitions between different states. Let $0 \leq i,j,s < K$, and*

$$\alpha_x \coloneqq \boldsymbol{\alpha}_{t-1}[x] \quad \boldsymbol{\alpha} \coloneqq \boldsymbol{\alpha}_{t-1}$$

$$\ell_x \coloneqq \mathcal{P}(\boldsymbol{y}[t] \,|\, \boldsymbol{q}[t] = x) \quad \boldsymbol{\ell} \coloneqq \mathcal{P}(\boldsymbol{y}[t] \,|\, \boldsymbol{q}[t])$$

$$x_{ij} \coloneqq \alpha_i \mathcal{A}_{ij} \ell_j,$$

so that

$$\boldsymbol{\alpha}_t[s] = \frac{\sum_i x_{is}}{\sum_{i,j} x_{ij}},$$
$$\widehat{\boldsymbol{\alpha}}_t[s] = \frac{x_{ss}}{\sum_j x_{jj}}.$$

Then we define the forward bias *of state s at position t as*

$$C_t := \frac{\widehat{\boldsymbol{\alpha}}_t[s]}{\boldsymbol{\alpha}_t[s]} = \frac{x_{ss}\sum_{i,j} x_{ij}}{\sum_{i,j} x_{jj}x_{is}}$$

Proposition 4.5.1 (Upper bound on forward bias)**.** *The forward bias is bounded as*

$$C_t \leq \frac{\sum_{i,j} x_{ij}}{\sum_j x_{jj}} = 1 + \frac{\sum_{i\neq j} x_{ij}}{\sum_j x_{jj}}.$$

In particular, if the transition probabilities are bounded as

$$\forall j : \mathcal{A}_{jj} \geq \delta,$$

$$\forall i \neq j : \mathcal{A}_{ij} \leq \epsilon,$$

the forward error is bounded as

$$C_t \leq 1 + \frac{\epsilon}{\delta}\left(\frac{\mathbf{1}\cdot\boldsymbol{\ell}}{\boldsymbol{\alpha}\cdot\boldsymbol{\ell}} - 1\right).$$

Proof.

$$C_t = \frac{x_{ss}\sum_{i,j} x_{ij}}{\sum_{i,j} x_{jj}x_{is}} = \frac{x_{ss}\left(\sum_j x_{jj} + \sum_{i\neq j} x_{ij}\right)}{\sum_{(i=s)}_j x_{jj}x_{ss} + \sum_{i\neq s}_j x_{jj}x_{is}} \tag{4.1}$$
$$\leq \frac{x_{ss}\left(\sum_j x_{jj} + \sum_{i\neq j} x_{ij}\right)}{x_{ss}\sum_j x_{jj}} = \frac{\sum_{i,j} x_{ij}}{\sum_j x_{jj}} \tag{4.2}$$
$$= \frac{\sum_j x_{jj}}{\sum_j x_{jj}} + \frac{\sum_{i\neq j} x_{ij}}{\sum_j x_{jj}} = 1 + \frac{\sum_{i\neq j} x_{ij}}{\sum_j x_{jj}} \tag{4.3}$$

Furthermore,

$$C_t \leq 1 + \frac{\sum_{i\neq j} x_{ij}}{\sum_j x_{jj}} \tag{4.4}$$
$$= 1 + \frac{\sum_{i\neq j} \alpha_i \mathcal{A}_{ij}\ell_j}{\sum_j \alpha_j \mathcal{A}_{jj}\ell_j} \tag{4.5}$$
$$\leq 1 + \frac{\epsilon\sum_{i\neq j} \alpha_i\ell_j}{\delta\sum_j \alpha_j\ell_j} \tag{4.6}$$

$$= 1 + \frac{\epsilon}{\delta}\left(\frac{\sum_{i,j}\alpha_i\ell_j - \sum_j \alpha_j\ell_j}{\sum_j \alpha_j\ell_j}\right) \tag{4.7}$$

$$= 1 + \frac{\epsilon}{\delta}\left(\frac{\sum_{i,j}\alpha_i\ell_j}{\sum_j \alpha_j\ell_j} - 1\right) \tag{4.8}$$

$$= 1 + \frac{\epsilon}{\delta}\left(\frac{\left(\sum_i \alpha_i\right)\left(\sum_j \ell_j\right)}{\sum_j \alpha_j\ell_j} - 1\right) \tag{4.9}$$

$$= 1 + \frac{\epsilon}{\delta}\left(\frac{\sum_j \ell_j}{\sum_j \alpha_j\ell_j} - 1\right) \tag{4.10}$$

□

Obviously, the bias is small for HMM with strong self-transitions, as well as α_s and ℓ_s approaching 1. For Gaussian HMM, these criteria are typically met for emissions with low variance and well-separated means whenever the sampled $\boldsymbol{\theta}^{(i)}$ approaches the true emitting $\boldsymbol{\theta}$, confirming the result of Mahmud & Schliep (2011) in those special cases. In general, due to the rearrangement inequality, the bound is tightest whenever the entries of $\boldsymbol{\alpha}$ and $\boldsymbol{\ell}$ are in the same order.

We conjecture that the forward bias is the reason for the observed fast convergence towards MPM segmentation. Consider the case of uniform self-transition and transition probabilities:

Proposition 4.5.2 (Bias towards largest forward variable)**.** *Let*

$$s = \arg\max_i \alpha_i$$

be the state with the largest forward variable at $t-1$. Let $\forall i : \mathcal{A}_{ii} = \delta$ and $\forall i \neq j : \mathcal{A}_{ij} = \epsilon$. Then

$$C_t \geq 1.$$

Proof. The forward bias of state s is

$$\begin{aligned} C_t &= \frac{x_{ss}\sum_{i,j} x_{ij}}{\sum_{i,j} x_{jj}x_{is}} = \frac{x_{ss}\sum_{i,j} x_{ij}}{\left(\sum_j x_{jj}\right)\left(\sum_i x_{is}\right)} = \frac{x_{ss}\left(\sum_j x_{jj} + \sum_{i\neq j} x_{ij}\right)}{\left(\sum_j x_{jj}\right)\left(x_{ss} + \sum_{i\neq s} x_{is}\right)} \\ &= \frac{x_{ss}\sum_j x_{jj} + x_{ss}\sum_{i\neq j} x_{ij}}{x_{ss}\sum_j x_{jj} + \left(\sum_j x_{jj}\right)\left(\sum_{i\neq s} x_{is}\right)} = \frac{\sum_j x_{jj} + \sum_{i\neq j} x_{ij}}{\sum_j x_{jj} + \sum_{\substack{j \\ i\neq s}} \frac{x_{jj}x_{is}}{x_{ss}}} \\ &= \frac{\sum_j x_{jj} + \sum_{i\neq j} \alpha_i\epsilon\ell_j}{\sum_j x_{jj} + \sum_{\substack{j \\ i\neq s}} \frac{\alpha_j\delta\ell_j\alpha_i\epsilon\ell_s}{\alpha_s\delta\ell_s}} = \frac{\sum_j x_{jj} + \epsilon\sum_{i\neq j} \alpha_i\ell_j}{\sum_j x_{jj} + \epsilon\sum_{\substack{j \\ i\neq s}} \alpha_i\ell_j\frac{\alpha_j}{\alpha_s}} \end{aligned}$$

$$= \frac{\delta \sum_j \alpha_j \ell_j + \epsilon \left(\sum_{j \neq s} \alpha_s \ell_j + \sum_{\substack{i \neq j \\ i \neq s}} \alpha_i \ell_j \right)}{\delta \sum_j \alpha_j \ell_j + \epsilon \left(\sum_{j \neq s} \frac{\alpha_j^2}{\alpha_s} \ell_j + \sum_{\substack{i \neq j \\ i \neq s}} \alpha_i \ell_j \frac{\alpha_j}{\alpha_s} \right)}$$

The claim follows since $\forall j : \alpha_s \geq \alpha_j$, so $\frac{\alpha_j}{\alpha_s} \leq 1$ and $\alpha_s = \frac{\alpha_s^2}{\alpha_s} \geq \frac{\alpha_j^2}{\alpha_s}$. □

In fact, a sufficient criterion can be established for the direction of the forward bias which does not involve the emission likelihoods.

Proposition 4.5.3 (Direction of forward bias)**.** *Let the number of states $K \geq 3$. Then for any relation $\asymp \in \{<, =, >\}$, the* forward bias *of state s*

$$C_t := \frac{\widehat{\boldsymbol{\alpha}}_t[s]}{\boldsymbol{\alpha}_t[s]} \asymp 1$$

if and only if

$$\sum_{\substack{i \neq s \\ j \neq s \\ i \neq j}} \ell_j \left(\alpha_s \mathcal{A}_{ss} \left(\alpha_i \mathcal{A}_{ij} + \frac{\alpha_s \mathcal{A}_{sj}}{K-2} \right) - \alpha_j \mathcal{A}_{jj} \left(\alpha_i \mathcal{A}_{is} + \frac{\alpha_j \mathcal{A}_{js}}{K-2} \right) \right) \asymp 0.$$

In particular, the relation holds if

$$\forall i \neq s, j \neq s, i \neq j : \quad \alpha_s \mathcal{A}_{ss} \mathcal{A}_{ij} \asymp \alpha_j \mathcal{A}_{jj} \mathcal{A}_{is} \tag{4.11}$$

$$\forall j \neq s : \quad \alpha_s^2 \mathcal{A}_{ss} \mathcal{A}_{sj} \asymp \alpha_j^2 \mathcal{A}_{jj} \mathcal{A}_{js}. \tag{4.12}$$

Proof. We rewrite $C_t \asymp 1$ as a comparison of numerator and denominator:

$$\sum_{i,j} x_{ss} x_{ij} \asymp \sum_{i,j} x_{jj} x_{is} \tag{4.13}$$

$$\sum_i x_{ss} x_{is} + \sum_{\substack{i \\ j \neq s}} x_{ss} x_{ij} \asymp \sum_i x_{ss} x_{is} + \sum_{\substack{i \\ j \neq s}} x_{jj} x_{is} \tag{4.14}$$

$$\sum_{\substack{i \\ j \neq s}} x_{ss} x_{ij} \asymp \sum_{\substack{i \\ j \neq s}} x_{jj} x_{is} \tag{4.15}$$

$$\sum_{j \neq s} x_{ss} x_{jj} + \sum_{\substack{j \neq s \\ i \neq j}} x_{ss} x_{ij} \asymp \sum_{j \neq s} x_{jj} x_{ss} + \sum_{\substack{j \neq s \\ i \neq s}} x_{jj} x_{is} \tag{4.16}$$

$$\sum_{\substack{j \neq s \\ i \neq j}} x_{ss} x_{ij} \asymp \sum_{\substack{j \neq s \\ i \neq s}} x_{jj} x_{is} \tag{4.17}$$

$$\sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} x_{ss}x_{ij} + \sum_{j\neq s} x_{ss}x_{sj} \asymp \sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} x_{jj}x_{is} + \sum_{j\neq s} x_{jj}x_{js} \tag{4.18}$$

On each side, there are $(K-1)^2-(K-1) = K^2-3K+2$ terms in the left and $K-1$ in the right sum. In order to match the index set, we replace each summand on the right by $K-2$ normalized copies, one for each $j \neq i \neq s$ on the left, yielding $(K-1)(K-2) = K^2-3K+2$ summands:

$$\sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} x_{ss}x_{ij} + \sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} \frac{x_{ss}x_{sj}}{K-2} \asymp \sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} x_{jj}x_{is} + \sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} \frac{x_{jj}x_{js}}{K-2}$$

We then get

$$\sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} \left(x_{ss}\left(x_{ij} + \frac{x_{sj}}{K-2}\right) - x_{jj}\left(x_{is} + \frac{x_{js}}{K-2}\right)\right) \asymp 0$$

Dividing by ℓ_s yields

$$\sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} \ell_j \left(\alpha_s\mathcal{A}_{ss}\left(\alpha_i\mathcal{A}_{ij} + \frac{\alpha_s\mathcal{A}_{sj}}{K-2}\right) - \alpha_j\mathcal{A}_{jj}\left(\alpha_i\mathcal{A}_{is} + \frac{\alpha_j\mathcal{A}_{js}}{K-2}\right)\right) \asymp 0$$

In particular, the proposition holds if

$$\sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} x_{ss}x_{ij} \asymp \sum_{\substack{i\neq s\\ j\neq s\\ i\neq j}} x_{jj}x_{is} \tag{4.19}$$

$$\sum_{j\neq s} x_{ss}x_{sj} \asymp \sum_{j\neq s} x_{jj}x_{js}, \tag{4.20}$$

which in turn holds especially if $\asymp$ holds for all matching summands,

$$\forall i \neq s, j \neq s, i \neq j: \quad x_{ss}x_{ij} \asymp x_{jj}x_{is} \tag{4.21}$$

$$\forall j \neq s: \quad x_{ss}x_{sj} \asymp x_{jj}x_{js}, \tag{4.22}$$

hence

$$\forall i \neq s, j \neq s, i \neq j: \quad \alpha_s\mathcal{A}_{ss}\ell_s\alpha_i\mathcal{A}_{ij}\ell_j \asymp \alpha_j\mathcal{A}_{jj}\ell_j\alpha_i\mathcal{A}_{is}\ell_s \tag{4.23}$$

$$\forall j \neq s: \quad \alpha_s s\mathcal{A}_{ss}\ell_s\alpha_s\mathcal{A}_{sj}\ell_j \asymp \alpha_j\mathcal{A}_{jj}\ell_j\alpha_j\mathcal{A}_{js}\ell_s, \tag{4.24}$$

and by canceling terms,

$$\forall i \neq s, j \neq s, i \neq j: \quad \alpha_s\mathcal{A}_{ss}\mathcal{A}_{ij} \asymp \alpha_j\mathcal{A}_{jj}\mathcal{A}_{is} \tag{4.25}$$

$$\forall j \neq s: \qquad \alpha_s^2 \mathcal{A}_{ss} \mathcal{A}_{sj} \asymp \alpha_j^2 \mathcal{A}_{jj} \mathcal{A}_{js}. \tag{4.26}$$

□

Let s be the state with the largest forward variable at t, i. e. $\forall j : \boldsymbol{\alpha}_t[s] \geq \boldsymbol{\alpha}_t[j]$. As shown before, for uniform δ and ϵ, those relations hold. However, the result also suggests a general tendency for $\widehat{\boldsymbol{\alpha}}_t[s] \geq \boldsymbol{\alpha}_t[s]$ if self-transitions and other transitions are each on the same orders. Since $\frac{\mathcal{A}_{ss}}{\mathcal{A}_{jj}} \geq \frac{\mathcal{A}_{is}}{\mathcal{A}_j}$ and $\frac{\mathcal{A}_{ss}}{\mathcal{A}_{jj}} \geq \frac{\mathcal{A}_{sj}}{\mathcal{A}_{js}}$ imply $C_t \geq 1$, approximate equality $\mathcal{A}_{ss} \approx \mathcal{A}_{jj} \approx \delta$ and $\forall i \neq j$: $\mathcal{A}_{ij} \approx \epsilon$ imply that the forward bias will be $C_t \gtrsim 1$.

We demonstrate this effect in Fig. 4.12: For 4 states, we simulated $\boldsymbol{\alpha}$ and $\boldsymbol{\ell}$ as random variates from a Dirichlet distribution $\mathrm{Dir}(10, 1, 1, 1)$, and the rows of $\mathcal{A}$ as a Dirichlet variate with parameter 100 for the diagonal entries and $\tau \in \{1, 25, 50, 100\}$ for the off-diagonal entries. We plot the elements of $\boldsymbol{\alpha}$ against those of $\widehat{\boldsymbol{\alpha}}$, where the data point is shown in red for the maximal entry α_s and blue for the others. Clearly, there is a tendency for α_s to yield $\widehat{\boldsymbol{\alpha}}_t[s] \geq \boldsymbol{\alpha}_t[s]$. As expected, the effect is less pronounced for lower τ, since those $\mathcal{A}$ have higher self-transition probabilities.

Since we have shown before that wavelet compression creates a block structure which is consistent with the locations of state transition in MPM segmentation, s is likely to indicate the maximum posterior margin state for the entire block starting at t. Having a forward bias greater than 1 means that s will be sampled with higher probability than under unbiased forward filtering. Also, due to the recursive nature of forward-variable computation, the error is expected to grow within a block, with the other forward variables approaching 0; this effect is illustrated in Fig. 4.13. Therefore, a wavelet compressed block is sampled as the MPM state under the parametrization of the current iteration, with a probability approaching 1 for large blocks.

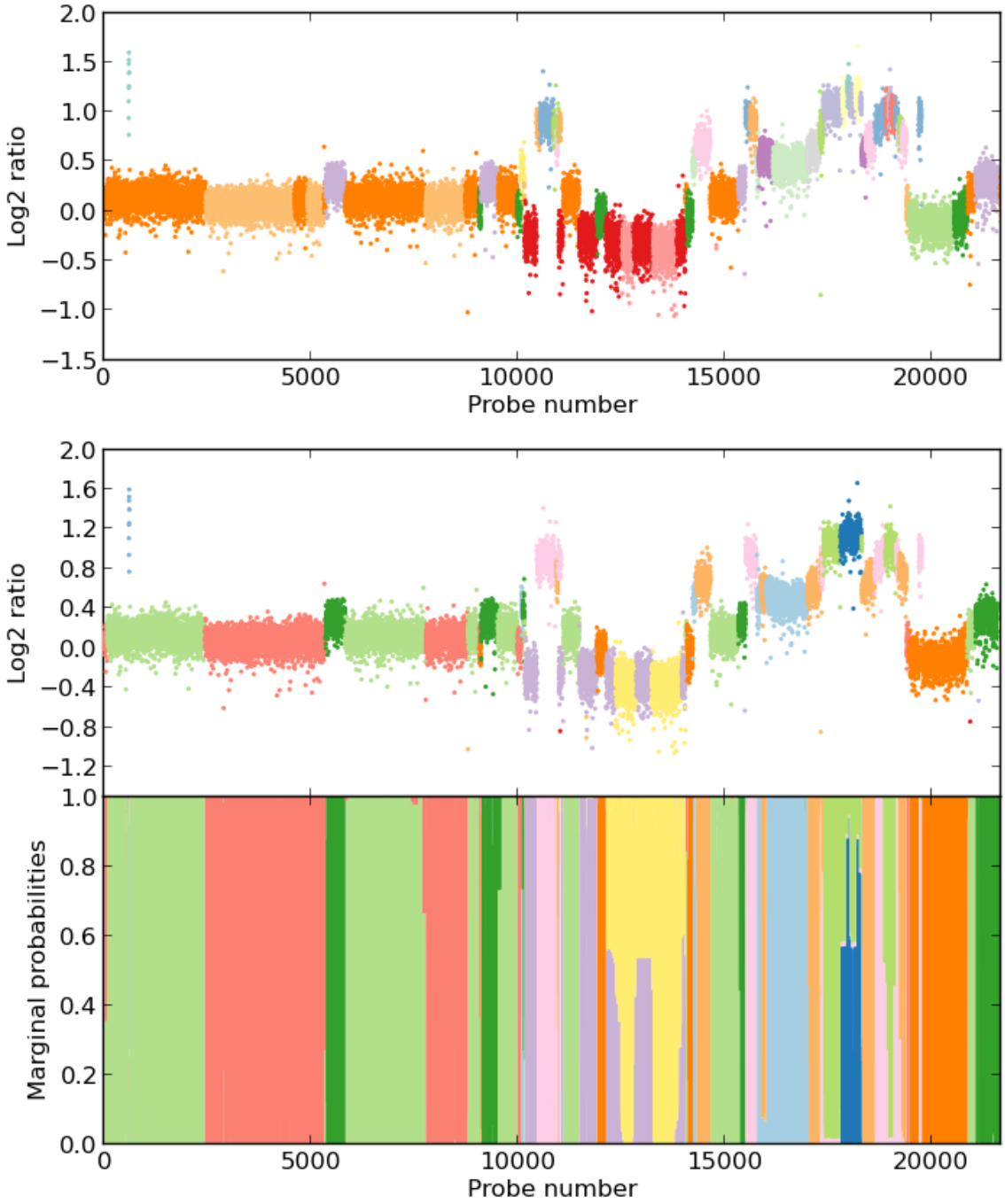


Figure 4.8: Copy number inference for chromosome 20 in invasive ductal carcinoma (21,687 probes). CBS creates a 19-state solution (top), however, a compressed 19-state HMM only supports an 11-state solution (bottom), suggesting insufficient level merging in CBS.

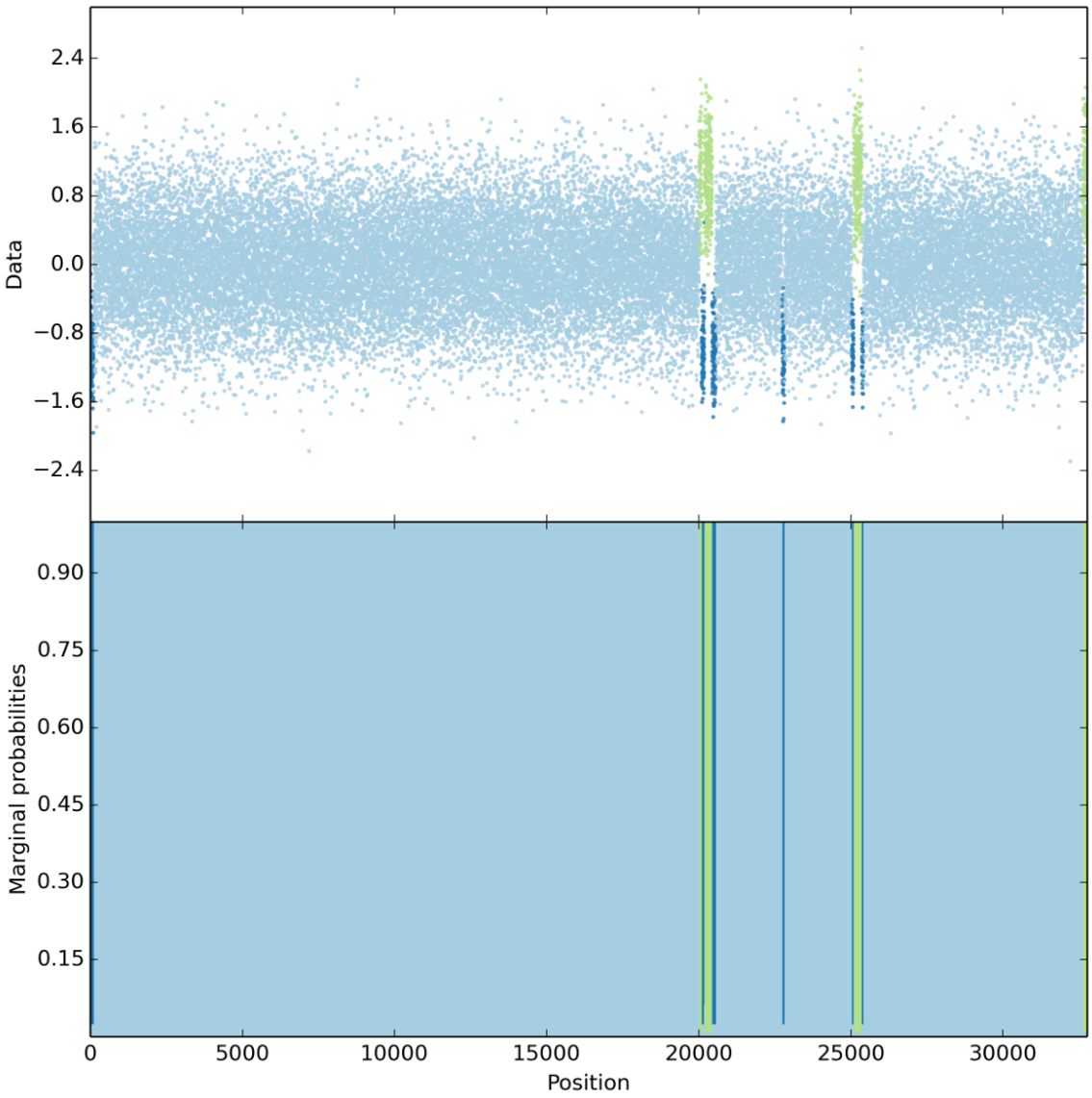


Figure 4.9: An example of a hard inference task from the simulations. Notice that the minority components have been correctly identified (green, dark blue), despite being very short and their means being contained well within the tails of the dominant emission distribution (light blue).

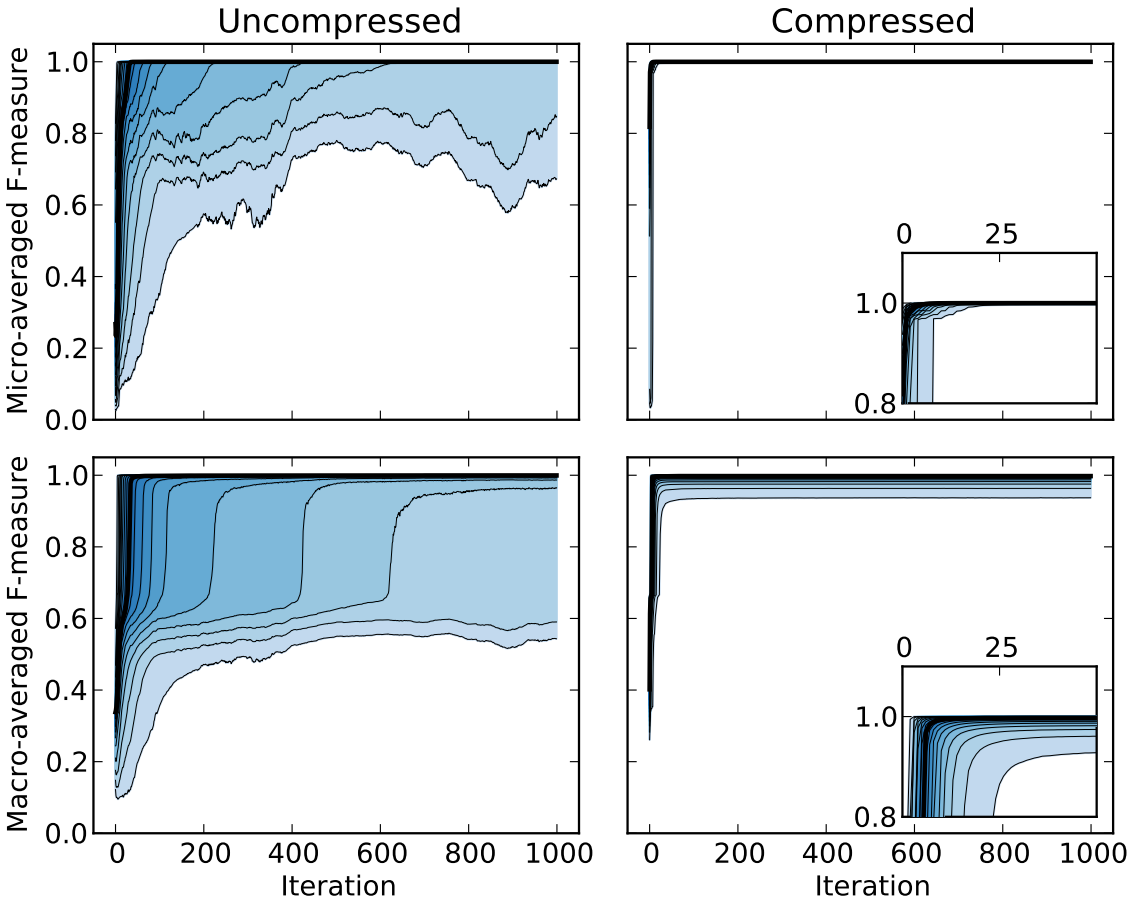


Figure 4.10: F-measures for simulation results. The median value (black) and quantile ranges (in 5% steps) of the micro- (top) and macro-averaged (bottom) F-measures (F_{mi}, F_{ma}) for uncompressed (left) and compressed (right) FBG inference, on the same 129,600 simulated data sets, using automatic priors. The x-axis represents the number of iterations alone, and does not reflect the additional speedup obtained through compression. Notice that the compressed HMM converges no later than 50 iterations (inset figures, right).

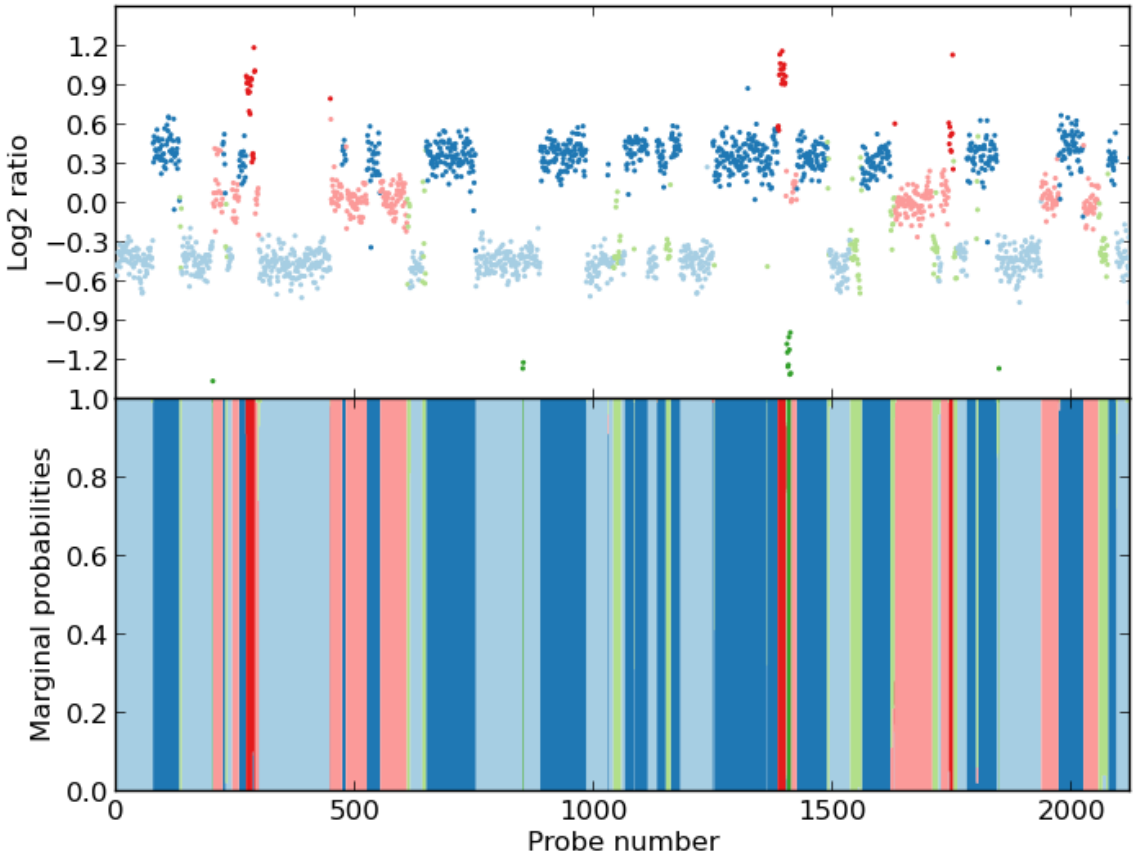
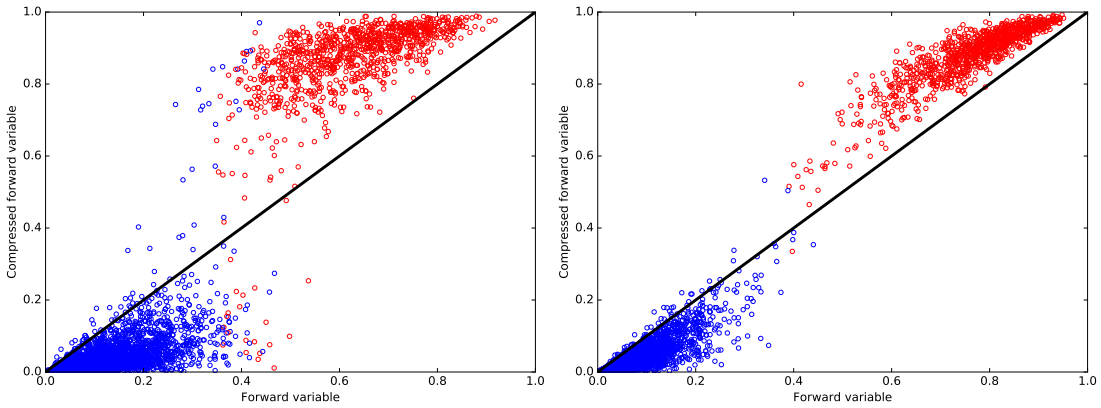


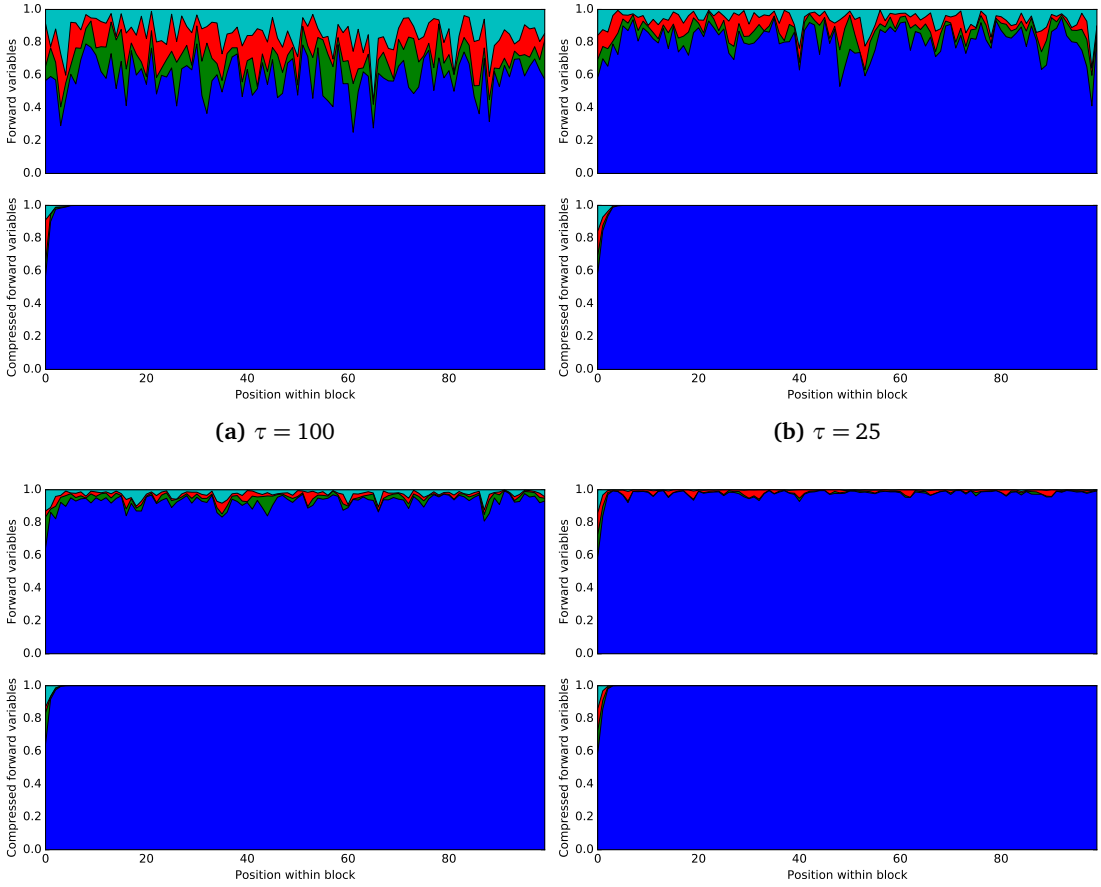
Figure 4.11: HaMMLET's inference of copy-number segments on T47D breast ductal carcinoma. Notice that the data is much more complex than the simple structure of a diploid majority class with some small aberrations typically observed for Coriell data.



(a) $\tau = 100$ **(b)** $\tau = 25$

(c) $\tau = 10$ **(d)** $\tau = 1$

Figure 4.12: Demonstration of the tendency of compressed forward variables to be biased towards the state with the largest marginal probability (red), and away from the others (blue). Variables on the diagonal are unbiased. Subplots are shown for different Dirichlet distributions $\mathrm{Dir}(100, \tau, \tau, \tau)$.



(a) $\tau = 100$ **(b)** $\tau = 25$

(c) $\tau = 10$ **(d)** $\tau = 1$

Figure 4.13: A demonstration of the increasing forward bias due to recursive computation of forward variables, for different Dirichlet weights τ. Regardless of the actual size of forward variables (upper subplots in each subfigure), the compressed forward variables quickly converge to 1 for the maximum state (blue).

Chapter 5

Algorithm engineering for big data applications

Having demonstrated the properties and theoretical background of HaMMLET in the previous chapters, we now address the issue of implementation beyond the prototype stage. While convergence and speed have been addressed, memory consumption remains problematic for genome-sized data. In this chapter, we describe our design decisions to scale HaMMLET to big-data applications. The main contributions in this chapter are as follows:

1. We show that the wavelet tree data structure yields under-compression, resulting in trellises which are too large.

2. The wavelet tree is also memory-inefficient: For a given T, it stores $2T - 1$ sufficient statistics and as many maximum coefficients. This is the exact number of distinct statistics required for arbitrary thresholds; however, the statistics stored are not the ones that are queried.

3. The wavelet tree was a monolithic data structure in which each node was used to store marginal state counts as well as a data point (leaves) or sufficient statistics (inner nodes). This implementation proved to be very wasteful, and did not scale to genomic data sizes.

4. To alleviate those issues, we designed specialized data structures for different functionalities previously contained in the wavelet tree: block boundaries are stored in a *breakpoint array*,

statistics for arbitrary blocks of size N are queried in $O(\ln N)$ time from a modified *integral array* of size T, and a queue-based implementation of *marginal records* is used to store and update run-length encoded counts for marginal state counts.

5. We show that for univariate data, the breakpoint array for Haar weights can be computed in-place in linear time.

6. For multivariate data, we show that the query time for each block is independent of dimensionality; once the block boundaries have been determined in $\ln N$ time, the statistics at dimension d can be queried in constant time, so iterating through all d dimensions of a block can be done in $d + \ln N$ time.

7. For multivariate data, a breakpoint array with Haar weights would require two arrays of size T to store the results of several Haar transforms in order to compute maxima across dimensions. Instead, we modify the lifting scheme for the Haar transform to process the input in a different order, allowing the data to be processed as an input stream without storing $O(T)$ temporary values. This yields a computation of coefficient maxima across dimensions in dT time and $T + d \operatorname{ld} T$ space.

5.1 Wavelet tree revisited

To motivate the need for a different implementation, let $T_{hg} := 3{,}500{,}000{,}000$, the approximate number of base pairs in the human genome. If wavelet weights and sufficient statistics are stored as IEEE-754 32-bit floating point numbers, then for Gaussian HMM we require a total of $5 \cdot 8T_{hg}$ bytes, or more than 130 GB RAM. It is possible to create optimal compression, i. e. one that does not introduce more breakpoints than there are discontinuities in the wavelet regression, but that would require storing both the absolute as well as the maximum subtree coefficients, adding another 26 GB.

Additionally, our original implementation stored the block sizes with the sufficient statistics, as the number of subtree leaves is not easily obtained from the BFS nor DFS pre-order index of a node in constant time, especially not if the number of leaves is not a power of 2 and the tree is truncated. For general implementations, the safe way to implement this is using an unsigned

integer type the size of a pointer on the target platform, and typically incurs at least another 8 byte twice per input position. Furthermore, using the pyramid algorithm, we require padding of the data to a power of two, and an auxiliary array for calculating the wavelet coefficients, which in the worst-case scenario almost quadruples the input size. In that case, the memory requirement increases to $80T$, or 26 GB for T_{hg}. These requirements can be avoided by using a modified version of an in-place Haar wavelet transform.

In addition, we previously used a tree-like layout to record the marginal counts for each block, which was wasteful and, in the worst case, could incur another $2TK$ unsigned integer variables large enough to count the number of iterations. Even for 2-byte integers, a 20-state HMM would incur another 131 GB for the human genome. In total, our original implementation of HaMMLET would require close to 400 GB RAM for a realistic model. Several implementation details such as suboptimal use of STL vectors, each adding 20 byte overhead, pushed the memory load to the terabyte range, resulting in excessive swapping, rendering the implementation useless for WGS data.

5.2 Dynamic block creation

Consider the following abstract data structure:

Definition 5.2.1 (Block generator)**.** *Let $\boldsymbol{b}$ be a vector of breakpoint weights. For a threshold λ, let $\boldsymbol{Y}_\lambda$ be a partition of $\boldsymbol{y}$ into blocks such that there is a block boundary between positions $t-1$ and t if $\boldsymbol{b}[t] \geq \lambda$. We call a data structure a* block generator *if it can, for any threshold λ, generate an ordered sequence of sufficient statistics that represents Y_λ. A block generator is called* compressive *if, for all λ, $\boldsymbol{b}[t] < \lambda$ implies that no breakpoint is created between $t-1$ and t. It is called* subcompressive *if for some λ such a superfluous block boundary is created. A block generator is called* space-efficient *if it stores no more than T sufficient statistics.*

This definition of a block generator implies that $\boldsymbol{Y}_{\lambda_1}$ is a subdivision of $\boldsymbol{Y}_{\lambda_2}$ if $\lambda_1 \leq \lambda_2$. For sufficiently small thresholds, we require sufficient statistics for each data point, hence any block generator implementation will have to store a minimum of T sufficient statistics. On the other hand, if all entries in $\boldsymbol{b}$ are unique, each breakpoint subdivides a block defined by a higher

threshold, and a simple induction argument shows that a block generator has to be able to generate $2T-1$ different blocks and their sufficient statistics: starting with a single block of size T and a sorted sequence of threshold values in $\boldsymbol{b}$, each threshold creates two new blocks by subdividing one block in the previous partition.

Proposition 5.2.1. *The wavelet tree is a subcompressive and memory-inefficient block generator.*

Proof. The wavelet tree is memory-inefficient as it stores $2T-1$ sufficient statistics. Wavelet compression is based on the fact that if all coefficients in a subtree are below the threshold, then so is the maximum coefficient in that subtree. The converse however is not true; if the maximum coefficient is above the threshold, then not all coefficients in the subtree have to be above threshold. Whenever the DFS traverses into a subtree, three breakpoints are introduced into the block structure, corresponding to the three discontinuities in the Haar wavelet represented by the subtree's root. This happens whenever the subtree contains any coefficient above threshold, regardless of whether the coefficient at the root itself is above threshold. Fig. 5.1 illustrates a simple counterexample to compressivity. □

It should be noted that, while the wavelet tree stores as many sufficient statistics as needed for T data points, the fact that it is subcompressive implies that the block structures it creates differ from those of a compressive block generator, and hence these are *not* the same $2T-1$ statistics that would occur in accross all block structures a compressive block generator would yield. Compressivity could be restored by keeping copies of the wavelet coefficients overwritten by the subtree maxima, which would potentially double the memory requirements for the coefficient array, and is thus not preferred for big data applications.

In order to provide an efficient implementation, we separate a block generator into two sub-structures: a *breakpoint array* to derive a sequence of start and end positions for blocks, and an *integral array* to query the sufficient statistics for each block.

5.2.1 Integral array for block statistics

Let a data structure $D(\boldsymbol{y})$ support the following query: given a start index s and an end index e, with $s < e$, return the sufficient statistics in the half-open interval $[s, e)$, i. e. $\sum_{i=s}^{e-1} \boldsymbol{T}(\boldsymbol{y}[i])$.

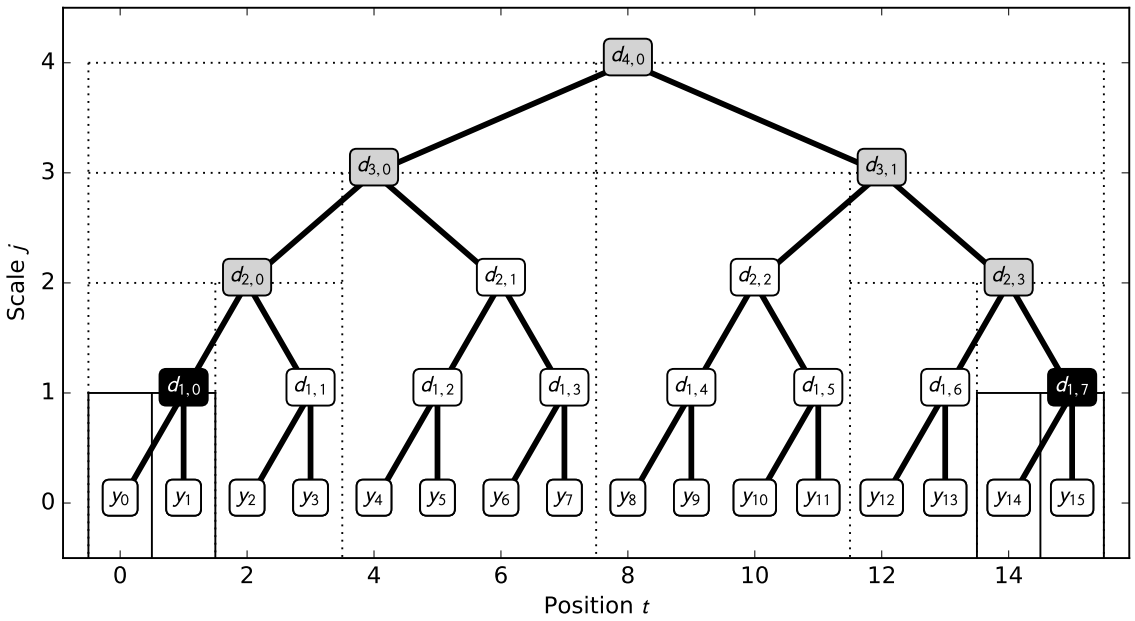


Figure 5.1: The wavelet tree data structure is subcompressive, as it induces additional breakpoints. In this example, coefficients with an absolute value above the threshold are denoted as black boxes. In the tree layout induces by the lifting scheme, the position of a coefficient equals that of the central discontinuity of its associated Haar wavelet. For instance, $\psi_{2,0}$ has positive support on $\boldsymbol{y}[0], \boldsymbol{y}[1]$, and negative support on $\boldsymbol{y}[2], \boldsymbol{y}[3]$, with its position 2 being the lowest position of its negative support. The positions of the block boundaries are indicated by thin solid vertical lines, connected to their respective tree node by vertical lines. In this example, $\left|d_{1,0}\right| > \lambda$, inducing block boundaries at 0, 1 and 2, and $\left|d_{1,7}\right| > \lambda$, inducing block boundaries at 14, 15 and 16, creating 5 blocks in total. By taking the maximum subtree coefficient, additional inner nodes contain values above threshold, indicated here by gray boxes, Traversing into subtrees below those nodes induces additional block boundaries, indicated here by dotted lines, at 2, 4, 8, 12 an 14. This yields a total of 8 blocks.

A trivial implementation of such a data structure would be to store the statistics of each input position, and then iterate through the array and calculate their cumulative sums between breakpoints. This is obviously costly for huge data, as it incurs $\Theta(N)$ time complexity for a block of size N. Constant-time queries could be made by pre-computing all T^2 statistics, which is obviously prohibitive for large data.

The basic idea for querying sufficient statistics comes from a simple data structure in image processing called a *summed-area table* or *integral image* (Lewis 1995), which is used to query the sum of a rectangular region in constant time. As its one-dimensional equivalent, let $\boldsymbol{v}$ be an *integral array* such that

$$\boldsymbol{v}[t] = \begin{cases} \boldsymbol{T}(0) & t = 0 \\ \sum_{i=0}^{t-1} \boldsymbol{T}(\boldsymbol{y}[t]) & t > 0. \end{cases}$$

3				7				11				15			
2	3			6	7			10	11			14	15		
1	2	3		5	6	7		9	10	11		13	14	15	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Figure 5.2: An illustration of an integral array $\boldsymbol{\nu}$, using cell size $c = 4$. Columns represent data positions, and contain all positions i which are added up and stored at $\boldsymbol{\nu}[t]$; for instance, $\boldsymbol{\nu}[9] = \sum_{i=9}^{11} \boldsymbol{T}(\boldsymbol{y}[i])$. The statistics of a block $[s,e)$ are obtained by adding $v[s]$, $\boldsymbol{\nu}[m]$ for all $s < m < e$, $m \equiv 0 \mod c$, and subtracting $\boldsymbol{\nu}[e]$ iff $e \not\equiv 0 \mod c$. For instance, block [3,10) is obtained as $\boldsymbol{\nu}[3] + \boldsymbol{\nu}[4] + \boldsymbol{\nu}[8] - \boldsymbol{\nu}[10]$, yielding $\sum_{t=3}^{9} \boldsymbol{T}(y[t])$.

For any arbitrary start and end positions s, e, the sufficient statistics of the block $[s,e)$ can be calculated in constant time as

$$\sum_{t=s}^{e-1} \boldsymbol{T}(\boldsymbol{y}[t]) = \left(\sum_{t=0}^{s-1} \boldsymbol{T}(\boldsymbol{y}[t])\right) - \left(\sum_{i=0}^{e-1} \boldsymbol{T}(\boldsymbol{y}[t])\right) = \boldsymbol{\nu}[e] - \boldsymbol{\nu}[s].$$

In contrast to image processing, where integral arrays are constructed over integer data, sufficient statistics require floating-point values for most distributions. Unfortunately, this incurs numeric problems at large data sizes. An IEEE 754 single-precision float has between 6 and 9 significant digits. Assuming that values for sufficient statistics are on the order of 1, the further back a data point is in $\boldsymbol{\nu}$, the more of its significant digits is used to store the sum. Neighboring entries will be similar or even equal, leading to catastrophic cancellation or even 0 for short segments. For instance, values above ~ 17 million are rounded to multiples of 2, so that even if each entry was 1.0, blocks of size 1 would be queried as 0.

To alleviate this, we subdivide $\boldsymbol{\nu}$ into non-overlapping *cells* of size c, and compute partial cumulative sums of sufficient statistics within each cell; for convenience, we compute these sums from high to low indices, see Fig. 5.2. It is then easy to see that

$$\sum_{t=s}^{e-1} \boldsymbol{T}(\boldsymbol{y}[t]) = \left(\sum_{j} \boldsymbol{\nu}[j]\right) - \boldsymbol{\nu}[e], \quad j \in \{s\} \cup \{i \,|\, s < i \leq e, i \equiv 0 \,(\mathrm{mod}\, c)\}$$

In our implementation, we used $c = 2^{16} = 65{,}536$.

Numerical issues Regardless of the data structure being used, an approach relying of sufficient statistics raises a general issue of numerical stability. In particular, for Gaussian emissions, updating the Normal-Inverse Gamma hyperparameter

$$\beta \leftarrow \beta + \frac{1}{2}\left(\Sigma_2 - \frac{\Sigma_1^2}{N} + \frac{N\nu}{N+\nu}\left(\frac{\Sigma_1}{N} - \mu\right)^2\right)$$

contains a numerically unstable term,

$$\Sigma_2 - \frac{\Sigma_1^2}{N}$$

akin to naive computation of the sample variance. As a safeguard, our implementation uses

$$\max\left\{0, \Sigma_2 - \frac{\Sigma_1^2}{N}\right\}$$

instead, in order to avoid non-positive values of β.

One advantage of using the wavelet tree was that the sufficient statistics were computed recursively bottom-up in a binary tree, an order which corresponds to a technique called *pairwise summation*. This is known to yield a relatively stable sum with at most $O(\epsilon\sqrt{\log T})$ roundoff-error, where ϵ is machine precision (Higham 1993).

Changing from a wavelet tree to an integral array trades this stability for a reduction in memory from $2T-1$ to T statistics. To counteract this effect, we use Kahan's summation algorithm (Kahan 1965) for cumulative sums within each cell, so that the expected summation error remains bounded independent of c.

5.2.2 Breakpoint array for block boundaries

In order to create a block generator, the integral array has to be supplemented with a data structure which yields start and end positions $s_k(\lambda)$, $e_k(\lambda)$ for subsequent blocks k. Since $e_k(\lambda) = s_{k+1}(\lambda)$, it suffices to implement an iterator over s_k for increasing k, where

$$s_0 = 0 \quad s_k = e_k(\lambda) = s_{k+1}(\lambda)$$

We use a simple array of pointers to facilitate these queries:

Definition 5.2.2 (Breakpoint array)**.** *Let $\boldsymbol{b} \in \mathbb{R}^T$ be a vector of breakpoint weights, and $\boldsymbol{p} \in \mathbb{Z}_+^T$ be a vector of pointers. A data structure $(\boldsymbol{b}, \boldsymbol{p})$ is called a* breakpoint array *of input data $\boldsymbol{y}$ if and only if*

$$\forall t < i < t + \boldsymbol{p}[t] : \boldsymbol{b}[t] > \boldsymbol{b}[i] \tag{5.1}$$

We call each interval $[t, \ldots, \boldsymbol{p}[t]-1]$ a stretch *at t. A breakpoint array is called* maximal *if for all T there exist no $n > \boldsymbol{p}[t]$ such that setting $\boldsymbol{p}[t]$ to n would still result in a valid breakpoint array.*

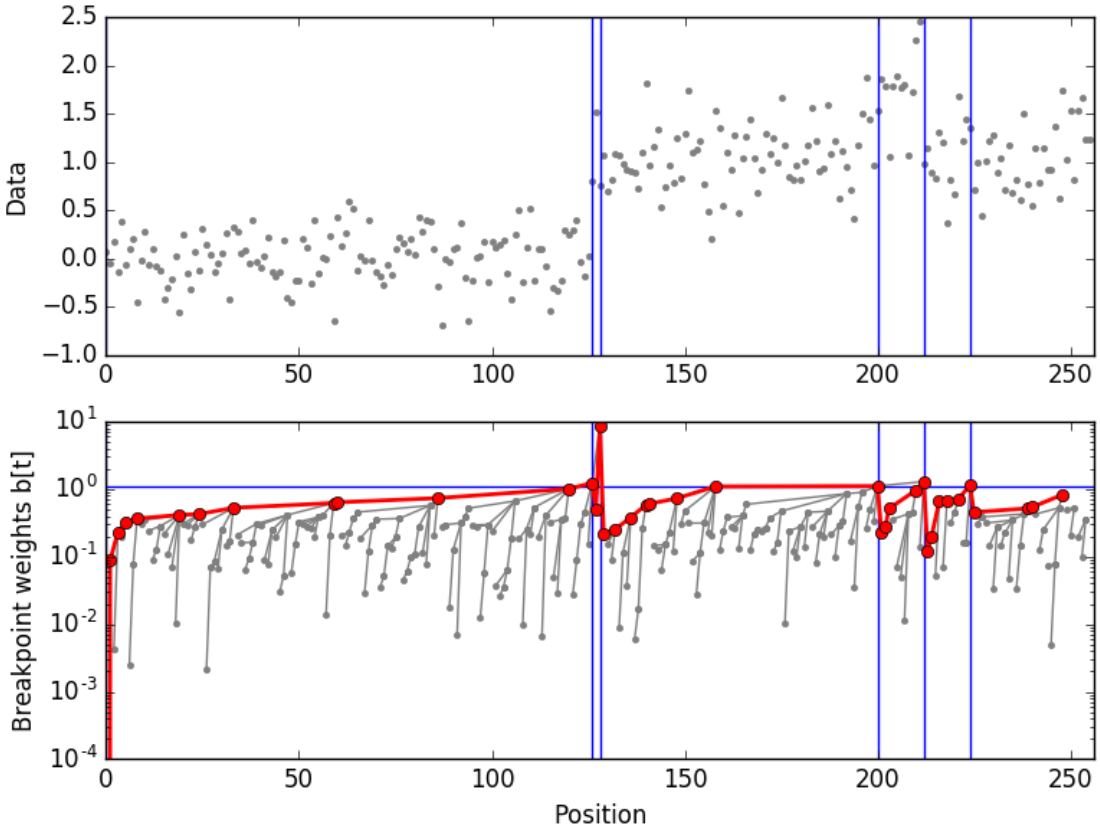


Figure 5.3: An example of generating blocks following pointers in a breakpoint array. The top figure represents the input data $\boldsymbol{y}$, the bottom figure represents the absolute wavelet coefficients, as well as the pointers (grey lines) and the path taken by the query (red). Whenever a value above the threshold (horizontal blue line) is found, a breakpoint is returned (vertical blue lines).

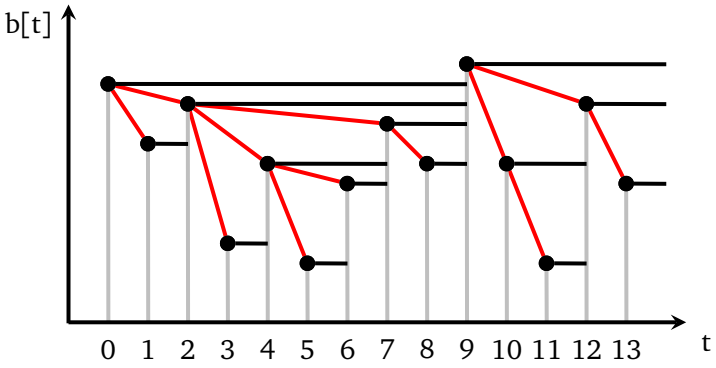


Figure 5.4: An example of a breakpoint array. Horizontal black lines represent the jumps indicated by the pointers, red lines represent stacks, and gray lines are for visual clarity. Upon processing index t, the index stack contains those elements obtained by following the red lines to the top left, starting at $b[t]$. Notice how this creates a sequence of right-to-left maxima.

Algorithm 2 Constructor of a maximal breakpoint array for a vector $\boldsymbol{b}$ of breakpoint weights, pointer array $\boldsymbol{p}$ and a maximum jump size m. S_t is a deque (double-ended queue).

```
1:  procedure BreakpointArrayConstructor(b, v, m)
2:      pushback(S_t, 0)                          ▷ make first position pending
3:      for t ← 1, ..., T − 1 do
4:          if |S_t| > 0 then
5:              if t − S_t[front] = m then        ▷ check distance of farthest element
6:                  p[S_t[front]] ← m              ▷ set farthest jump pointer
7:                  popfront(S_t)                  ▷ mark as processed
8:          while |S_t| > 0 do                    ▷ go through stack to find pending elements
9:              if b[S_t[back]] ≤ b[t] then       ▷ pending elements with smaller weights
10:                 reduceStack()                 ▷ set pending pointers and statistics
11:             else
12:                 break                         ▷ rest of stack has larger weights
13:         push(S_t, t)                          ▷ make current position pending
14:     t ← T
15:     while |S_t| > 0 do                        ▷ all remaining elements point to one-past-the-end
16:         reduceStacks()
17:
18: function reduceStack()
19:     i ← S_t[back]                             ▷ get closest pending index i
20:     p[i] ← t − i                              ▷ set its pointer to the distance to current index
21:     popback(S_t)                              ▷ remove index from stack
```

Proposition 5.2.2. *Algorithm 2 constructs a breakpoint array in linear time $O(T)$.*

Proof. A linear-time algorithm to calculate the pointers to the next element at least as large as the current one is well established in algorithmic folklore. It is modified here to use the distance to that element instead of a direct pointer (line 20, which would normally read $\boldsymbol{p}[i] \leftarrow t$). The stack is changed to a deque to accommodate the inclusion of a maximum jump size m. The front of the deque is popped and its pointer set whenever it is m positions away, which happens at most T times. □

For each t, $\boldsymbol{p}[t]$ points to the beginning of next stretch. Within each stretch, the highest breakpoint weight is located at its first position; when searching for weights below a given threshold λ, once the first weight is found to be below λ, all others can be safely ignored, leading to a simple query (Algorithm 3, demonstrated in Fig. 5.3). Given a position $e_k(\lambda)$ of block k under threshold λ, the next position $e_{k+1}(\lambda)$ can be found in $O(\ln N)$ expected time, where $N = e_k(\lambda) - s_k(\lambda)$. In order to derive the query complexity, we require the following result:

Algorithm 3 Given a breakpoint weight threshold λ, and a breakpoint position s_i this method computes the next breakpoint position s_{i+1} where $\boldsymbol{b}[s_{i+1}] > \lambda$ It returns true if such a signal exists, and false otherwise to indicate that the iterator is finished.

```
1:  procedure BreakpointIterator(s_i, λ)
2:      if s_i ≥ T then                                  ▷ no more blocks left
3:          return false
4:      s_{i+1} ← s_i + 1                                ▷ set potential end to next position
5:      while s_{i+1} < T do                             ▷ determine the end of the current block
6:          if b[s_{i+1}] < λ then                       ▷ current weight below threshold
7:              s_{i+1} ← s_{i+1} + p[s_{i+1}]           ▷ skip the following lower weights
8:          else
9:              break                                    ▷ higher weight than at block start determines block end
10:     N ← s_{i+1} − s_i                                ▷ set current block size
11:     return true
```

Proposition 5.2.3 (Left-to-right maxima (Lovász 1993; Knuth 1997))**.** *For a vector $\boldsymbol{x}$, let $\boldsymbol{x}[t]$ be called a* left-to-right maximum *of $\boldsymbol{x}$ iff $\forall i < t : \boldsymbol{x}[i] < \boldsymbol{x}[t]$. Let $m_{\boldsymbol{x}}$ count the number of left-to-right maximal elements in $\boldsymbol{x}$. For a random permutation of $\boldsymbol{x}$ with $|\boldsymbol{x}| = N$ elements,*

$$\mathbb{E}[m_{\boldsymbol{x}}] = \sum_{i=1}^{N} \frac{1}{N} \to \ln N \qquad \text{as} \qquad N \to \infty. \tag{5.2}$$

Due to symmetry, the same result holds for minima and right-to-left extrema.

Proposition 5.2.4. *For a block of size N, the expected query complexity of Algorithm 3 is $O(\log N)$ in a maximal breakpoint array.*

Proof. Since the breakpoint array is maximal, following pointers in $\boldsymbol{p}$ to find the block end creates a sequence of left-to-right maxima. For a block of size N, starting at t, there are $M := N-2$ elements in $I := [t+1, \ldots, t+N-1]$ which can appear in any order, and the claim follows from Eq. (5.2). □

Proposition 5.2.5. *Assume all elements in $\boldsymbol{b}$ have different values. Then the maximum expected stack size in Algorithm 2 is at most $\ln T$.*

Proof. Assume $m = \infty$. An element at t is pushed whenever there exists an index j on the stack such that $\forall i = j, \ldots, \text{top} : \boldsymbol{w}[i] < \boldsymbol{w}[t]$. Given the smallest such j, the stacks are popped until $\text{top} = j-1$, and $\boldsymbol{w}[j-1] > \boldsymbol{w}[t]$. Therefore, the stack contains the right-to-left minima of $\boldsymbol{w}[1{:}t]$ after pushing index t, and the claim follows from Eq. (5.2) for $t = T$. For any $m < \infty$, the front of the deque gets popped, thus only decreasing the stack size. □

For T_{hg}, the expected maximum stack size is < 22, a negligible overhead. We noticed that, for noisy data, most entries in $\boldsymbol{p}$ are much smaller than T, and using pointer-sized integers such as `size_t` in C++ (typically 8 byte on 64-bit systems), would be wasteful. Instead, we use a 2-byte unsigned integer type to accommodate jumps up to $m = 65{,}536$. The resulting breakpoint array is not maximal anymore, but maintains its space-efficiency and compressivity, as Algorithm 3 does not require maximality. The query overhead is minimal in practice; even in case of a single block for genome sized data, $\frac{T_{hg}}{65{,}536} < 54$.

Proposition 5.2.6. *Using a breakpoint array to iterate over block boundaries, which are then used to query sufficient statistics from an integral array yields a compressive and space-efficient block generator.*

Proof. To prove that the breakpoint array is indeed a valid iterator over block boundaries, Algorithm 3 provides a constructive description of how a block is created for an arbitrary threshold λ. Its correctness is shown by induction: Assume that the start position s_i is a valid block boundary, so $\boldsymbol{b}[s_i] \geq \lambda$ or $s_i = 0$. The block is determined by finding the next start position s_{i+1}, which can be found by setting $s_{i+1} \coloneqq s_i + 1$, and incrementing until $\boldsymbol{b}[s_{i+1}] \geq \lambda$. Instead of incrementing by 1, Eq. (5.1) guarantees we can increment s_{i+1} by $\boldsymbol{p}[s_{i+1}]$ without missing a breakpoint (lines 4–9). The block size $N = s_{i+1} - s_i$ is then easily calculated (line 10). Since no superfluous breakpoint is introduced, the data structure is compressive, provided that the sufficient statistics can be generated correctly. Since the first block starts at $s_0 = 0$, the claim follow by induction. As the number of sufficient statistics in a breakpoint array is T, it is also space-efficient. □

On first sight, the query complexity of $O(\log N)$ appears to be suboptimal compared to the constant-time queries in the original wavelet tree. This, however, is misleading. We have shown that the wavelet tree implements a suboptimal compression in the sense that it introduces additional block boundaries which do not correspond to discontinuities in the Haar transform. For a true block of size N, it creates on the order of $\ln N$ different blocks. Hence, not only does it take $\ln N$ such queries to cover the entire block, it also increases the size of the trellis and hence both space and runtime requirements. The breakpoint array thus has the same amortized complexity as the wavelet tree, while creating better compression.

5.2.2.1 Haar breakpoint weights

Having established a data structure to iterate over blocks for any given compression level, we now define a vector $\boldsymbol{b}_H$ of breakpoint weights for the Haar wavelet transform, and derive a simple algorithm to calculate it from $\boldsymbol{y}$. We need the following definitions:

Definition 5.2.3 (Maxlet arrays)**.** *For $b^{\pm}_{j,k} \in [0, T)$, let*

$$\boldsymbol{b}_M\left[b^{\pm}_{j,k}\right] = \begin{cases} \infty & t = 0 \vee b^{-}_{j,k} \geq T \\ \left|\left\langle \boldsymbol{\psi}_{j,k}, \boldsymbol{y} \right\rangle\right| & t > 0 \vee b^{-}_{j,k} < T \end{cases}$$

be a vector of absolute Haar wavelet coefficients, called a univariate maxlet array*. For multivariate data, the* multivariate maxlet array *is the pointwise maximum of the univariate maxlet arrays for each data dimension.*

Notice that the univariate maxlet array corresponds to the absolute values of Haar wavelet transforms, generalized for arbitrary T: If T is a power of 2, it contains the absolute values of detail coefficient of the Haar wavelet transform with its first entry (the scaling coefficient) replaced by ∞. For other T, it consists of a concatenation of such segments of decreasing size. For instance, if $T = 22$, the transform concatenates 3 arrays whose sizes are powers of two ($22 = 16 + 4 + 2$). Later in this chapter, we provide two algorithms:

1. The *univariate maxlet transform* computes the maxlet array for arbitrary data sizes T in-place and in linear time. In Fig. 5.5, the dashed lines and circular markers in the upper subplot illustrate the necessary updates, and the white markers in the lower subplot correspond to the result.

2. The *multivariate maxlet transform* computes the multivariate maxlet array; instead of computing d univariate maxlet transforms separately, using a space of $2T$ (one array of size T for calculating the transform at the current dimension, another for storing the maxima), we show that this transform can be computed using only $T + d \operatorname{ld} T$ space and $O(d\,T)$ time. For $d = 1$, the result is the same as the univariate maxlet transform.

In a maxlet array, each value at t corresponds to the (maximum of d) detail coefficients for which the central discontinuity $b^{\pm}_{j,k} = t$. In order to decide whether or not a wavelet regression

under a threshold λ has a discontinuity at t, all wavelets with $b^+_{j,k} = t$ and $b^-_{j,k} = t$ have to be taken into account. The breakpoint weights are thus computed as follows:

Definition 5.2.4 (Haar breakpoint array)**.** *For $b^\pm_{j,k} \in [0, T)$, let*

$$\boldsymbol{b}_H\left[b^\pm_{j,k}\right] = \begin{cases} \infty & t = 0 \vee b^-_{j,k} \geq T \\ \max_{j',k'} \left\{ \left| \left\langle \boldsymbol{\psi}_{j',k'}, \boldsymbol{y} \right\rangle \right| \,\middle|\, b^\pm_{j,k} = b^+_{j',k'} \vee b^\pm_{j,k} = b^-_{j',k'} \right\} & t > 0 \vee b^-_{j,k} < T \end{cases}$$

be a vector of breakpoint weights. A breakpoint array initialized with these weights is called a Haar breakpoint array.

Here, the entry at each position t is set to the largest absolute coefficient for all wavelets which have their left, central or right discontinuity at t, as long as the wavelet for which $b^\pm_{j,k} = t$ has full support in $[0, T)$. The reasoning behind this definition is that if T is a power of 2, then this breakpoint array introduces the same breakpoints as wavelet compression for the same threshold, with probability 1; a breakpoint is only potentially hidden in cases where $b^-_{j,k} = b^+_{j,k+1}$; on $\mathbb{R}$, this has probability measure 0. For other data sizes, there are wavelets which have their central discontinuity within the data, but have incomplete support. Instead of using one of several padding schemes, we assume those breakpoints to have infinite weight, which introduces up to $\operatorname{ld} T$ additional breakpoints.

If the emissions are multivariate with dimension d, the set of block boundaries is the union of block boundaries across all dimensions. In other words, the corresponding weights for the Haar breakpoint array can easily be calculated as the per-position maxima of across the d such weight vectors, as given by the multivariate maxlet array.

To derive Haar breakpoint weight from any maxlet transform, we introduce the *Haar boundary transform*, which performs the necessary maximum computations in-place and in linear time $O(T)$. In Fig. 5.5, the solid lines in the upper subplot illustrate the computation of maxima, and the black markers in the lower subplot correspond to the result.

5.2.2.2 In-place univariate maxlet transform

In-place calculation of $\boldsymbol{b}_M$ requires an in-place generalization of the Haar wavelet transform for arbitrary-size data. The pyramid algorithm used in our original approach was obviously not

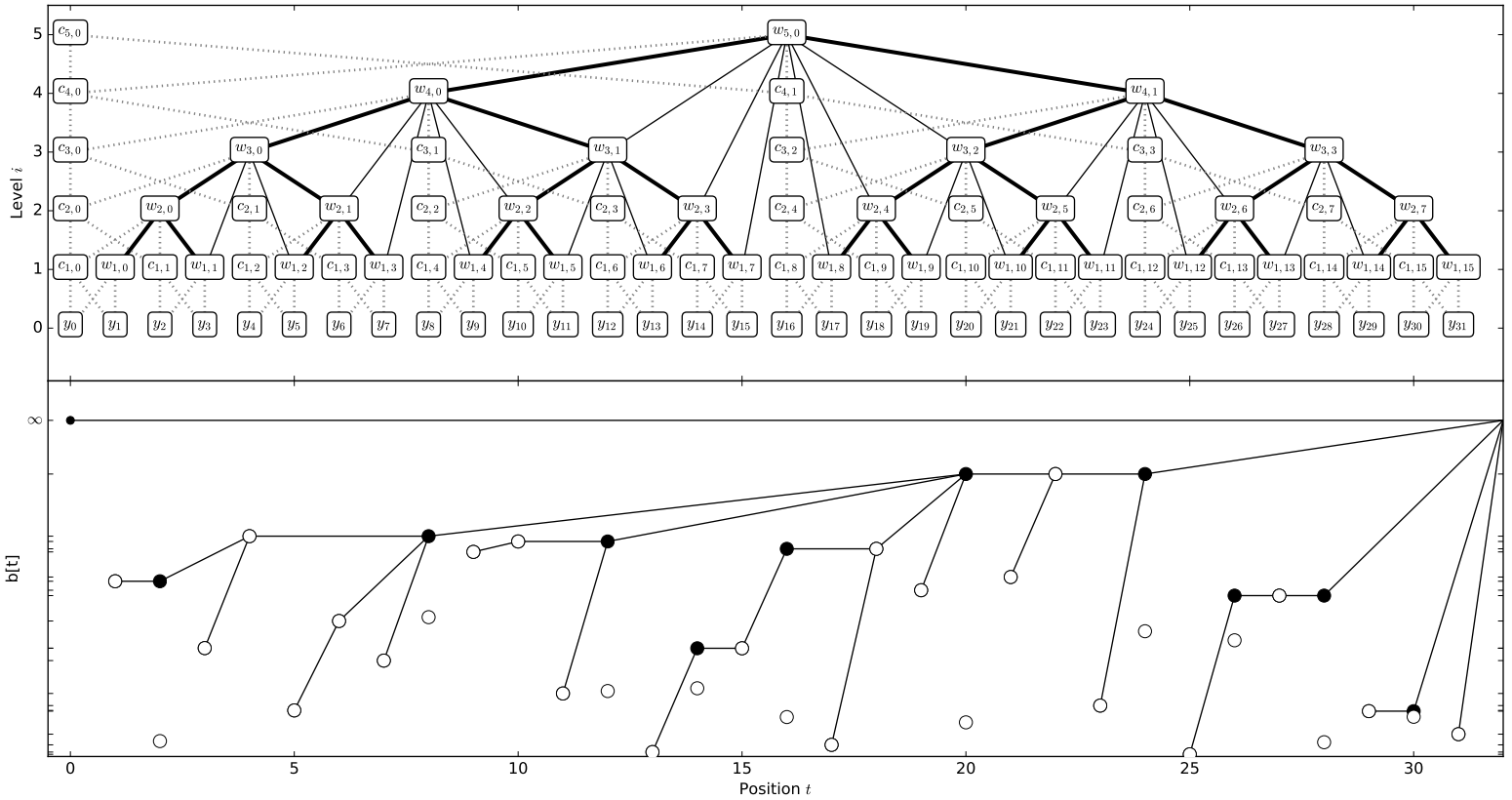


Figure 5.5: Illustration of the various algorithms necessary to create the Haar breakpoint array in-place. The top figure represents the transformation of an input array $\boldsymbol{y}$ at level 0 into various other forms. The terms c_{ij} and w_{ij} represent values associated with the scale and detail coefficients of the wavelet transform, respectively. The wavelet tree (bold lines) represents the nested nature of the support intervals: the horizontal position of $\psi_{i,j}$ represents the position t of central discontinuity $\boldsymbol{b}_{i,j}^{\pm}$ of $\boldsymbol{\psi}_{i,j}$, and its vertical position represents the resolution level i. The support interval for each wavelet corresponds to all descendants at level 0. The tree nodes contain the output arrays of the various transforms. Dotted lines indicate the recursive relations in the lifting scheme, as used by the in-place Haar wavelet transform and the maxlet transforms. The solid lines (including tree edges) indicate the dependencies in the Haar boundary transform.

in-place, as it required padding of the input array to a power of two, as well as an additional array of size T for temporary storage. In order to derive an in-place algorithm to calculate $\boldsymbol{b}_M$, we use a more recent in-place calculation of the Haar wavelet transform based on the lifting scheme (Algorithm 4), due to Sweldens (1995, 1998). It is based on the following recursions:

$$c_{j,k} \coloneqq \begin{cases} \boldsymbol{y}[k] & j = 0 \\ \sum_{t=b^+_{j,k}}^{b^-_{j,k}-1} \boldsymbol{y}[t] = \sum_{t=b^+_{j,k}}^{b^\pm_{j,k}-1} \boldsymbol{y}[t] + \sum_{t=b^\pm_{j,k}}^{b^-_{j,k}-1} \boldsymbol{y}[t] = c_{j-1,2k} + c_{j-1,2k+1} & j > 0 \end{cases}$$

$$d_{j,k} \coloneqq \frac{1}{\sqrt{2^j}} \left(\sum_{t=b^+_{j,k}}^{b^\pm_{j,k}-1} \boldsymbol{y}[t] - \sum_{t=b^\pm_{j,k}}^{b^-_{j,k}-1} \boldsymbol{y}[t] \right) = \frac{1}{\sqrt{2^j}} \left(c_{j-1,2k} + c_{j-1,2k+1} \right)$$

These relations are illustrated in Fig. 5.5 using dotted edges, with $d_{j,k} = w_{j,k}$ and $c_{0,k} = y_k = \boldsymbol{y}[k]$. By storing $c_{j,k}$ at index $b^+_{j,k}$ and $d_{j,k}$ at index $b^\pm_{j,k}$, they derive an in-place algorithm which never overwrites $d_{j,k}$ once it is calculated (Algorithm 4). Notice that each detail coefficient $d_{j,k}$ is stored at the position $b^\pm_{j,k}$ corresponding to the central discontinuity in their corresponding wavelet, and that this corresponds to an in-order DFS layout of the wavelet tree without the leaves corresponding to the input data, with the leftmost leaf at index 1 (Fig. 5.5, bold lines); the tree is created from the leaves up, and from left to right.

Algorithm 4 In-place Haar wavelet transform for power-of-2 data sizes (Sweldens 1995, modified for expositional purposes).

```
1:  procedure HaarTransform(y)
2:      T ← |y|                                  ▷ number of data points
3:      for j ← 1, ..., ld T do                  ▷ iterate over levels, bottom-up
4:          N ← 2^j                              ▷ support size of ψ_{j,k}
5:          s ← 1/√N                             ▷ normalization constant
6:          for k ← 0, ..., T/N − 1 do           ▷ process elements on level j from left to right
7:              L ← Nk                           ▷ left index
8:              R ← N(k + 1/2)                   ▷ right index
9:              y[L] ← y[L]                      ▷ copy c_{j−1,2k}
10:             y[R] ← y[R]                      ▷ copy c_{j−1,2k+1}
11:             y[L] ← y[L] + y[R]               ▷ calculate c_{j,k}
12:             y[R] ← s(y[L] − y[R])            ▷ calculate d_{j,k}
13:     return y
```

We first provide a generalization of the in-place Haar wavelet transform to arbitrary data sizes (Algorithm 7).

Proposition 5.2.7. *$\boldsymbol{b}_M$ can be computed in-place and in linear time $O(T)$.*

Algorithm 5 Given emission data $\boldsymbol{y}$, compute the univariate maxlet transform, i. e. the absolute detail coefficients of the Haar wavelet transform for arbitrary data sizes T.

```
1:  procedure MaxletTransform(y)
2:      T ← |y|                                   ▷ number of data points
3:      for j ← 1, ..., ⌈ld T⌉ do                 ▷ iterate over levels, bottom-up
4:          N ← 2^j                               ▷ support size of ψ_{j,k}
5:          s ← 1/√N                              ▷ normalization constant
6:          for k ← 0, ..., ⌈T/N⌉ − 1 do          ▷ process elements on level j from left to right
7:              L ← Nk                            ▷ left index
8:              R ← N(k + 1/2)                    ▷ right index
9:              if R < T then
10:                 y[L] ← y[L]                   ▷ copy c_{j−1,2k}
11:                 y[R] ← y[R]                   ▷ copy c_{j−1,2k+1}
12:                 y[L] ← y[L] + y[R]            ▷ calculate c_{j,k}
13:                 y[R] ← s|y[L] − y[R]|         ▷ calculate |d_{j,k}|
14:             else
15:                 y[L] ← ∞                      ▷ force breakpoint for incomplete support
16:     return y
```

Proof. $\boldsymbol{b}_M$ can be considered as the concatenation of a minimum number of maxlet transforms $\boldsymbol{b}_M^p$ of decreasing sizes which are all powers of 2. Note that $\boldsymbol{b}_M[t] = \infty$ whenever $\boldsymbol{b}_M^p[0]$. Since the support size for any Haar wavelet is a power of two, an infinite value indicates that the value associated with this position has incomplete support across the data. Assume w.l.o.g. that T is a power of two and hence the tree is complete. The Haar wavelet transform can be implemented in-place in linear time (Sweldens 1995), yielding a DFS in-order layout of the wavelet tree, with the first entry representing the sum of all data points. The first element can be set to ∞ and the other values can be set to their absolute value in a second pass. Alternatively, the transform can be calculated for arbitrary data sizes in one pass (Algorithm 7, which is a modification of the lifting scheme with checks for incomplete support intervals). □

5.2.2.3 Multivariate maxlet transform

Naively, the maxlet transform for multivariate data can be computed using $2T$ space, by successively computing the univariate maxlet transform for each dimension d in an array of size T, and continuously updating the maxima across dimensions in a second array of the same size. Here, we derive an alternative version requiring only an array of size T as well as a stack of size $O(d \operatorname{ld} T)$. Essentially, it is an adaptation of the lifting scheme, which changes the order of com-

putation (Algorithm 6). For an inner node in the tree of detail coefficients to be computed, all its descendants have to be known. In a streaming setting, the leaves arrive ordered by t. Therefore, the lifting scheme has to be changed from bottom-up BFS to DFS post-order, necessitating the use of a stack.

The algorithm maintains the following invariant: when processing the detail coefficient d_{jk} at position p, the stack S contains the unnormalized scaling coefficients (i. e. sums of data segments of size 2^j) $c_{j-1,2k+1}$ for all dimensions at the top D elements, followed by the set of unnormalized scaling coefficients for $c_{j-1,2k+1}$ at the D positions below. Using a stack that allows random access, such as the `vector` class in C++, detail coefficients the maximum d_{jk} across all data dimensions can be computed from the $2D$ top elements of the stack, using the lifting recursions and the level-specific normalization factor n (Algorithm 6, Line 16–Line 25). Afterwards, by adding the top D entries in the stack to the next D entries element-wise and popping the top D elements, the stack contains the dimension-wise sums for the joint support ranges, i. e. the unnormalized scaling coefficients to compute the left parent of the current node (the parent index and checks whether such a parent exist for the current position is maintained in the index p and a bitmask m reflecting the structure of the tree). For instance, if the stack contains $c_{2,2}$ and $c_{2,3}$, replacing $c_{2,2}$ by $c_{2,2} + c_{2,3}$ and popping $c_{2,3}$ yields $c_{3,1}$ on top of the stack, so $d_{4,0}$ can be calculated, since the elements below the stack top are the scaling coefficients $c_{3,0}$, see Fig. 5.5. Since the behavior of the stack mimics that of a DFS traversal, it never gets larger than $d \operatorname{ld} T$.

5.2.2.4 Haar boundary transform

For each central discontinuity $b^{\pm}_{j,k}$, there are $j-1$ wavelets at lower levels $j' < j$, which have their left discontinuity at that position ($b^{+}_{j',k'} = b^{+}_{j,k}$). The same is true for right discontinuities, so that each position is the maximum of $2j-1$ absolute wavelet coefficients. In Fig. 5.5, these relations are indicated in that each node in the tree is transformed into the maximum of itself as well as direct descendants in lower levels indicated by solid lines (including tree edges).

We will later show that despite those dependencies, the Haar breakpoint array can be calculated in-place in linear time for arbitrary-sized input data $\boldsymbol{y}$.

The linear running time for maximum calculation can be established non-constructively: The central discontinuity of a wavelet at level j is shared with j wavelets for which it is the left

Algorithm 6 Computes the univariate or multivariate maxlet transform, i. e. for each position t and D Haar wavelet transforms of size T the maximum of the D absolute wavelet coefficients at position t is computed. The input is assumed to come from a data stream F, sorted by (t,d), $0 \le t < T$, $0 \le d < D$. The algorithm requires $T + D \operatorname{ld} T$ space and time.

```
1:  procedure StreamingMaxletTransform(F, D)
2:      Allocate empty vector b                          ▷ Contains resulting maxlet transform
3:      Allocate empty stack S                           ▷ Contains unprocessed data sums
4:      t ← 0                                            ▷ Current data index
5:      d ← 0                                            ▷ Current dimension index at position t
6:      while F not empty do
7:          Get next element from F and push it to S
8:          d++                                          ▷ Dimension index of next element
9:          if d = D then                                ▷ Finished reading all dimensions for this position
10:             d ← 0
11:             Append ∞ to b                            ▷ Coefficients are ∞ for incomplete support
12:             p ← t                                    ▷ Index of detail coefficient in upward-left path (DFS)
13:             m ← 1                                    ▷ Bit mask for accessing left parent node, if it exists
14:             n ← 1/√2                                 ▷ Wavelet normalizer
15:             while (p ⩓ m) > 0 do                     ▷ p is a left parent in tree
16:                 c ← 0                                ▷ Maximum detail coefficient across dimensions
17:                 L ← |S| − 2D                         ▷ Lower stack index
18:                 R ← L + D                            ▷ Upper stack index
19:                 for D iterations do
20:                     c ← max{c, n|S[L] − S[R]|}       ▷ Update maximum at j
21:                     S[L] ← S[L] + S[R]               ▷ Store sum for next level
22:                     L++                              ▷ Next dimension for lower stack elements
23:                     R++                              ▷ Next dimension for upper stack elements
24:                 b[p] ← c                             ▷ Store maximum absolute detail coefficient
25:                 n ← n/√2                             ▷ Update normalizer for next level
26:                 Pop D elements from S                ▷ Top now contains subtree sum
27:                 p ← p − m                            ▷ Move to potential left parent
28:                 m ← 2m                               ▷ Shift bit mask
29:             t++
```

discontinuity, and likewise for the right, hence each wavelet coefficient at level j needs to be updated by $2j$ coefficients at lower levels. There are $\frac{T}{2^{j+1}}$ coefficients at level j, hence the total number of updates is

$$\sum_{j=1}^{\operatorname{ld} T} \frac{T}{2^{j+1}} 2j \le 2T$$

since the infinite series

$$\lim_{\operatorname{ld} T \to \infty} \sum_{j=1}^{\operatorname{ld} T} \frac{j}{2^j} = 2$$

is monotonically increasing.

Algorithm 7 Given a maxlet transform $\boldsymbol{d}$, for each position t compute the maximum absolute coefficient of all wavelets which have a discontinuity at t, in-place and in linear time.

procedure HAARBOUNDARYTRANSFORM($\boldsymbol{d}$)
  $\boldsymbol{d}[0] \leftarrow \infty$ ▷ force breakpoint before first element
  for $j \leftarrow \lfloor \operatorname{ld} T \rfloor, \ldots, 1$ **do** ▷ iterate over levels, top-down
    $N \leftarrow 2^j$ ▷ support size of wavelet t level j
    for $k \leftarrow 0, \ldots, \lceil \frac{T}{N} \rceil - 1$ **do** ▷ process elements on level j from left to right
      $t \leftarrow N(k + \frac{1}{2})$ ▷ index of central discontinuity $\boldsymbol{b}^{\pm}_{j,k}$ of $\boldsymbol{\psi}_{j,k}$
      $n \leftarrow \frac{N}{2}$ ▷ distance to left and right discontinuity
      if $t < T$ **then**
        $L \leftarrow t - n$ ▷ index of left discontinuity
        $\boldsymbol{d}[L] \leftarrow \max\{\boldsymbol{d}[L], \boldsymbol{d}[t]\}$ ▷ consider left discontinuity
        $R \leftarrow t + n$ ▷ index of right discontinuity
        if $R < T$ **then**
          $\boldsymbol{d}[R] \leftarrow \max\{\boldsymbol{d}[R], \boldsymbol{d}[t]\}$ ▷ consider right discontinuity
  return $\boldsymbol{d}$

At each position, left and right discontinuities of multiple wavelets have to be taken into account to calculate the maximum, except for positions between trees, which are already ∞. Two versions of this approach can be used: one can traverse through the tree in a BFS fashion, and update $b^{\pm}_{\ell,k}$ from the $b^{+}_{\ell',k'}$ and $b^{i}_{\ell',k'}$ at lower levels $\ell' < \ell$. Alternatively, the same updates can be performed traversing top-down through levels ℓ', updating weights at higher levels for left and write discontinuities. Each wavelet is considered at most twice for updating a wavelet on a higher level. Since higher-level weights are updated from lower levels, and the traversal is top-down, no weights are overwritten prematurely and the algorithm is in-place.

5.3 Compressed marginal records

Definition 5.3.1 (Marginal records)**.** *Let $t \in [0, \ldots, T)$, $s_{\max}$ the largest state sampled during FBG, and $s \in [0, \ldots, s_{\max}]$. A* marginal record *is a data structure which allows to store and query the number of times state s was observed at data index t.*

Our previous solution to recording marginal state counts was closely tied to the wavelet tree data structure in that it stored count vectors in the nodes of the tree. This was inefficient for a number of reasons: firstly, as discussed earlier, the number of nodes is larger than necessary. Secondly, the memory for each of these nodes has to be allocated. If the tree is implemented

as a flat array of size $2T$, the allocation requirements for k states are $2Tk$, even though a lot of nodes will potentially not contain any counts, and even those that do will contain zeros for many states. Such a preallocation approach also requires the number of states to be known in advance, and precludes further extensions to priors on the state number such as the Dirichlet Process.

We had therefore opted for a second approach, in which counts are dynamically allocated in each wavelet tree node using C++ `vectors`, such that the i-th position contains the count of state i. Vectors were dynamically increased in size to accommodate counts for the highest numbered state in each iteration. However, even with this approach, there was considerable overhead for enabling such dynamic allocation, in this case 20 B per node for holding start, end, and reserved memory size for vectors. For a billion data points, this alone results in 40 GB of additional RAM.

This could be somewhat alleviated by using a run-length encoding (RLE) approach, in which state counts are recorded for each compression segment and stored along with the segment length, illustrated by the right column of Fig. 5.6.

Dynamic compression however complicates the use of run-length encoding for marginals. At each new iteration, a different block structure is created, which requires existing RLE segments to be split into multiple parts, each of which will have counts for a different state added. This could be solved trivially using a linked list implementation, in which new segments are inserted with the appropriate updates of its neighbors size. This approach however has two disadvantages. Firstly, maintaining the pointers in the list is wasteful in terms of memory. Second, in order to hold the state count record, either a fixed array needs to be allocated, which is wasteful if the number of states is large, as it will contain mostly zero-entries, or some kind of dynamic data structure which encodes a compressed version of the state counts is required, which incurs additional memory due to additional householding variables. On the other hand, states could be stored in a resizable array like C++'s `vector`. It could be increased in size to hold the new number of run-length segments, and move existing items so as to free the necessary insertion positions. While this would only incur linear-time overhead, it requires the insert positions to be known with random-access. However, we do not explicitly store the block sizes for a state sample, as this incurs large memory usage, and obtain them by iterating over the breakpoint array instead. In this case, every entry after a new segment position would have to be moved

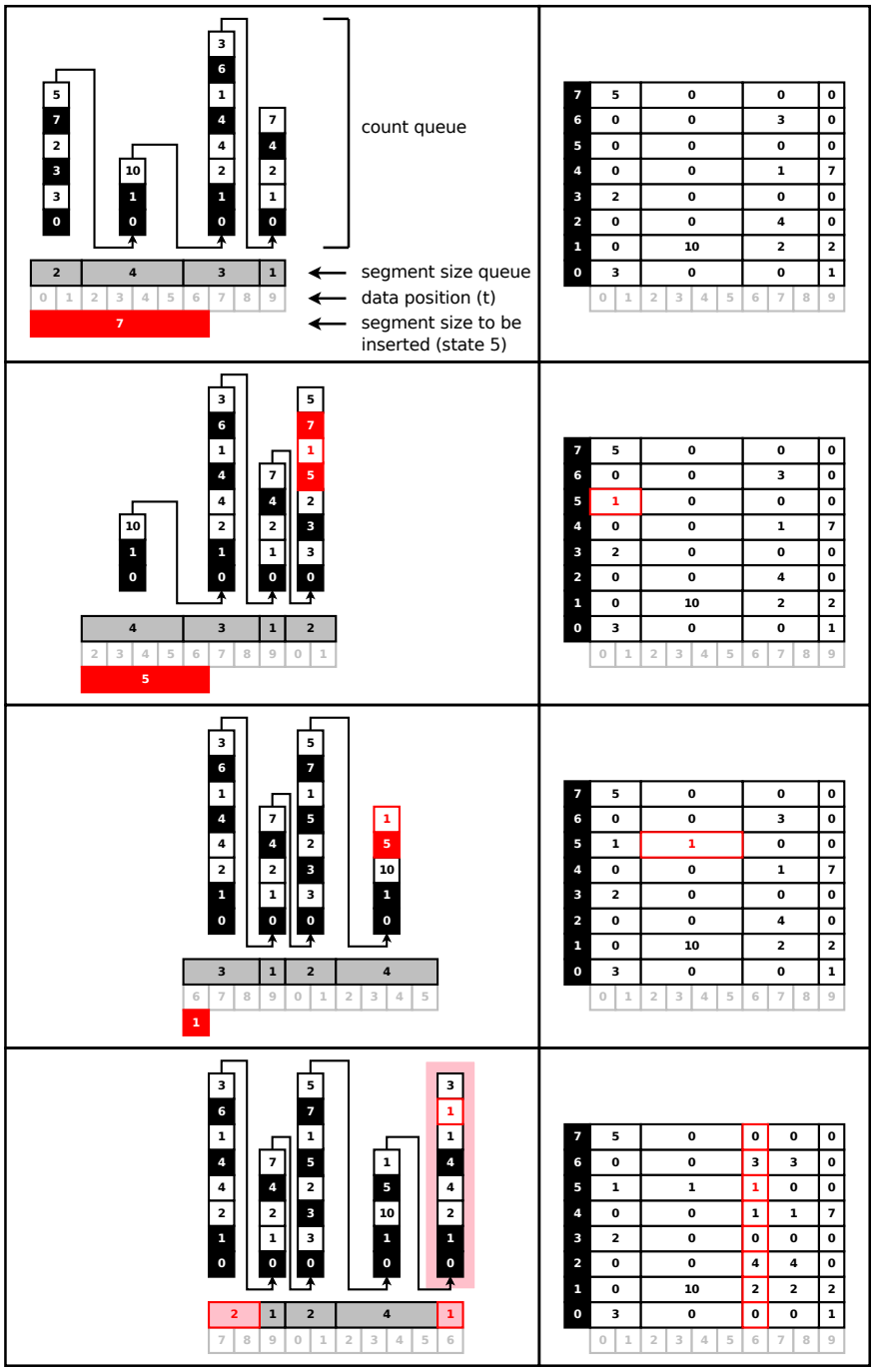


Figure 5.6: A small three-step example of recording marginal counts using Algorithm 8. Starting at position $t = 0$, 7 observations of state 5 are inserted. In the count queue, black boxes indicate that state counts of zero have been skipped; those numbers encode the next higher state that has a non-zero count. White boxes indicate the counts for the state. For instance, the right-most part of the count queue in the top subfigure is stored as $(0, -1, -2, 4, -7)$, indicating that there is 1 count for state 0, 2 counts for state 1, and 7 counts for state 4. The segment starts at position $t = 9$, and has a length of 1. Note that 0 is used to mark the start of a new segment. Each segment has a total of 10 counts already recorded. Arrows indicate contiguous elements in the count queue. With every iteration, a segment is moved to the back with the new state count included. Note that in the last iteration, the segment $t = 6, \ldots, 8$ is split. After finishing this step, the next count would be recorded starting at position $t = 7$. Notice how each run of zeros in the state queue is represented by a single number, thus allowing for arbitrarily large state indices without much overhead.

further back in the vector, so that the reallocation complexity for vector size n is on the order of $O(n^2)$ in each FBG iteration. Furthermore, this approach requires the marginals to be stored as individual data structures with the same memory overhead described above. If, on the other hand, the encoding was sequential so as to fit into a single array, determining the positions of the segments to be moved is challenging.

We developed an encoding for marginal records that stores counts sequentially in a vector of integers in a highly compressed fashion with small overhead. Adding records for run-length encoded state sequences is performed using a queue with iterator access to its front elements, such as implemented by `deque`, and requires a single pass over the state records and is therefore linear. The memory overhead is 2 bytes per segment, plus one bit for every 32 integers.

Encoding for marginal counts for a single position is performed using a sequence $\boldsymbol{c}$ of signed integers. A negative number is used to store the counts for a state. The state $s(i)$ of a position i is recursively defined as

$$s(0) = 0,$$

$$s(i) \coloneqq \begin{cases} s(i-1) & c[i-1] < 0 \\ \boldsymbol{c}[i-1] & c[i-1] > 0 \end{cases}.$$

Positive entries are called *index values*. We further require that all index values must be in strictly increasing order, and that no unnecessary index is used, i. e. we require

$$\forall \boldsymbol{c}[i] > 0 : s(i-1) + 1 < \boldsymbol{c}[i].$$

In other words, runs of states having observed counts are represented as runs of negative numbers, and runs of zero-counts are represented as a single number indicating the state label of the next higher state with non-zero counts. For instance, the count vector

$$(2, 0, 0, 8, 1, 4, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 0)$$

would be encoded as

$$(-2, 3, -8, -1, -4, 9, -5),$$

and the corresponding states are

$$(0, 1, 3, 4, 5, 6, 9),$$

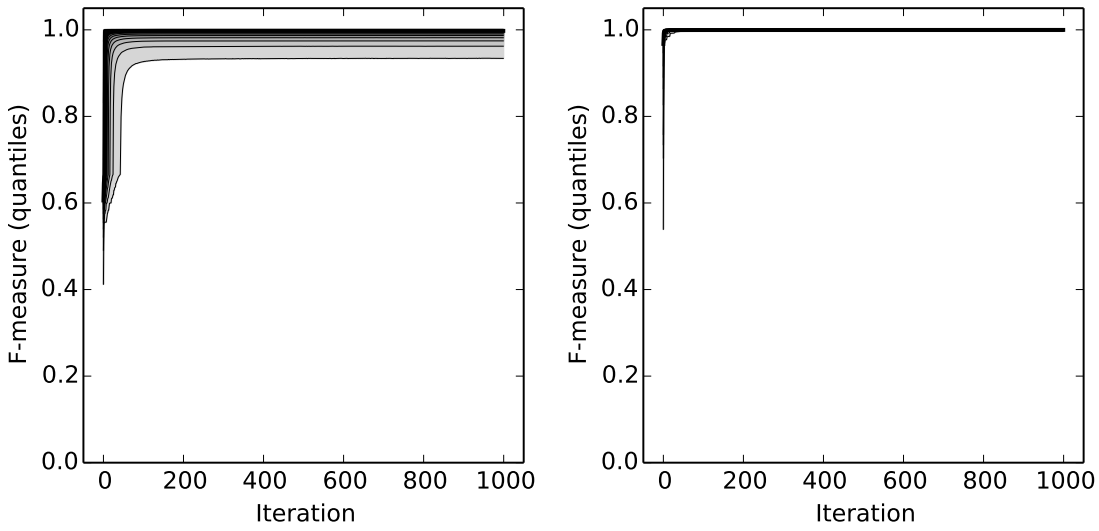


Figure 5.7: Micro- (left) and macro-averaged F-measures for the improved implementation of HaMMLET, using integral array, breakpoint array and the streaming maxlet transform. For comparison with the previous implementation results, see Fig. 4.10 on page 80.

though 1 and 6 are somewhat inconsequential as they have no counts associated with them; note that the decision to use negative signs for counts instead of index values is arbitrary in principle, but leads to using fewer negations in the implementation. In settings where quick convergence is expected, the number of zeros is expected to be high, leading to good compression under this scheme. In general, assume that the marginals contain M distinct segments after running FBG, and the HMM has S states. Then, the queue can contain no more than $(2S+1)M$ entries: for each segment, one zero to mark the beginning of a segment, and up to one positive and negative value per state. If the number of latent HMM states is limited to S, then there can be no more than S non-zero entries per segment. Hence, for reasonably high compression ratios, this amounts to small memory usage. For instance, at a compression ratio of 300 for a human genome at base-level resolution and 10 latent HMM states, marginal records using 2-byte signed integers require less than 234 MB. In practice, not every segment will contain 11 values, due to fast convergence, and the numbers get even smaller. Compared to the storage requirements of the block generator, this is negligible.

5.4 Evaluation

Changing the compression scheme raises the question of its effects on the inference quality. In particular, a stronger compression allowing for fewer path changes might have two competing effects. On the one hand, the decreased freedom in sampling the state sequence might result in slower mixing. On the other hand, assigning a state other than the MPM state to a block might yield posteriors which could prolong the convergence of said state parameters to their MPM value. In order to evaluate which of these effects is stronger, we rerun our previous simulations, using the exact same files created earlier and deposited on Zenodo. Fig. 5.7 shows the micro- and macro-averaged F-measures for our new implementation. Results are virtually identical, thought the macro-averaged F-measure appears to converge slightly slower for the lower quantiles. The slower mixing effect appears to be slightly stronger, yet negligible. Benchmarking results comparing the prototype and improved implementations of HaMMLET can be found in Section 6.4

Algorithm 8 Append N observations of state s to the marginal records.

```
1: procedure AddMarginalRecord(N, state)
2:     bool done ←false
3:     s ←0                                              ▷ State at current index
4:     while N > 0 do                                    ▷ Consume entire length of insert block
5:         s ←0                                          ▷ Assume lowest state
6:         done ←false                                   ▷ Remember if insert was successful
7:         for t ←0 t < countQ.size() ++t do
8:             entry ←countQ [t]                         ▷ Entry at current position
9:             if entry = 0 then                         ▷ All entries for this segment have been consumed
10:                if ¬ done then                        ▷ Count not registered yet
11:                    if s < state then                 ▷ Skipped over states?
12:                        countQ.push( state )          ▷ Add state index
13:                    countQ.push( −count )             ▷ Append count as negative number
14:                countQ.push( 0 )                      ▷ Append 0 to mark end of this segment
15:                break
16:            if done then                              ▷ Count was successfully registered
17:                countQ.push( entry )                  ▷ Keep appending all entries for this segment
18:                continue
19:            if entry > 0 then                         ▷ Entry denotes state of next entry
20:                if state < entry then                 ▷ Count must be inserted here
21:                    if s < state then                 ▷ Skipped over states?
22:                        countQ.push( state )          ▷ Add state index
23:                    countQ.push( -count )             ▷ Append count as negative number
24:                    if state + 1 < entry then         ▷ Skipping states until next count?
25:                        countQ.push( entry )          ▷ Add state index
26:                    done ←true                        ▷ Mark count insertion as successful
27:                else
28:                    countQ.push( entry )
29:                s ←entry                              ▷ Update state for next entry
30:            else                                      ▷ Entry is negative count or current state
31:                if s = state then                     ▷ Reached target state to be counted
32:                    countQ.push( entry − count )      ▷ Add count
33:                    done ←true                        ▷ Mark count insertion as successful
34:                else
35:                    countQ.push( entry )              ▷ Append entry without further action
36:                s++                                   ▷ Update current state
37:        if N < sizeQ.front() then                     ▷ Residual front segment remains
38:            sizeQ.push( N )                           ▷ Assign its size to new segment
39:            sizeQ.front() ←sizeQ.front() − N          ▷ Decrease size to remainder
40:            break
41:        else                                          ▷ Front segment was completely absorbed
42:            sizeQ.push( sizeQ.front() )               ▷ Associate its size with new segment
43:            N ←N- sizeQ.front()                       ▷ Set insert size to its remainder
44:            sizeQ.pop()                               ▷ Remove empty segment's size
45:            while countQ.front() ≠ 0 do               ▷ Remove all entries for mpty segment
46:                countQ.pop()
47:            countQ.pop()                              ▷ Remove segment separator
```

Chapter 6

Application to WGS data

In this chapter, we demonstrate the applicability of HaMMLET to a real-world research question. We show that CNV candidates inferred using our methods are enriched for gene annotations consistent with a working hypothesis that CNVs play a role in the *domestication syndrome*.

6.1 CNV as a genetic basis for domestication effects in rats

The domestication of a handful of animal species, starting in the early Holocene, has played a crucial role in the development of complex human societies (Diamond 1998). While we have learned a great deal about when and where animal domestication occurred, the genetic changes that underlie the phenotypic differences between domestic animals and their wild progenitors remain relatively unknown. It has been observed that domestic animal species tend to share a suite of behavioral, physiological and morphological traits that are absent or rarely observed in their wild progenitors (Darwin 1868; Wilkins, Wrangham & Fitch 2014). These traits include changes in pigmentation, craniofacial anatomy, hormonal levels, seasonal reproduction cycles and increased docility (Sánchez-Villagra, Geiger & Schneider 2016). This suite of changes is referred to as the "domestication syndrome". A long-standing question in evolutionary biology is whether these convergent changes are the result of genetic drift, artificial selection by humans for each individual trait, or pleiotropic effects of selection for a few or even a single trait. A proponent of the pleiotropy hypothesis, i.e. that genes under selection for behavioral traits of tameness and aggression influence other, seemingly unrelated phenotypes, was the Academician Dmitry K.

Belyaev. He hypothesized that selection for tameness at the start of the domestication process had pleiotropic effects that explained many of the features of the domestication syndrome. To test his hypothesis, he began a program of experimental domestication of the silver fox (Vulpes vulpes) in 1959 in Novosibirsk, Siberia. Foxes obtained for fur farms were selectively bred for their behavioral response to an approaching human. One line of foxes was bred for tame behavior towards humans, while a control line was selected for a fearful and aggressive response towards humans, to maintain the wild-type behavior despite being maintained in captive conditions. After just a few generations of selective breeding, the tame line began to show many of the traits associated with the domestication syndrome, including changes in pigmentation, morphology and behavior (Belyaev 1969; Trut, Plyusnina & Oskina 2004; Trut, Oskina & Kharlamova 2009).

The same experimental setup of artificially selecting two lines, one for tame and one for fearful and aggressive behavior towards humans, was also repeated by the same research group in the brown Norway rat (Rattus norvegicus) with similar results (Albert et al. 2008). These results seem to confirm Belyaev's hypothesis that selection for tameness alone could explain many of the features of the domestication syndrome. However, the specific genetic changes that underlie these changes remain unknown. Knowledge of the genetic variants that have been selected in these lines could lead to mechanistic insights into the domestication process. Genomic structural variants are of particular interest, as they are known to have played a role in the adaptation of other domestic animals (Axelsson et al. 2013), and structural variants that affect multiple functional genomic loci are one possible explanation for the rapid response to selection observed in these lines. To address this issue we analyzed whole-genome data that was generated from multiple individuals from the tame and aggressive lines of rats. For an overview of the experimental process described below, see Fig. 6.1.

6.2 Sample origins and data generation

DNA samples were obtained from two rat lines originating from a shared wild source population, subsequently maintained in isolation and divergently selected for ∼70 generations for their behavioral response to humans. 20 samples were obtained from the tame line, which has been

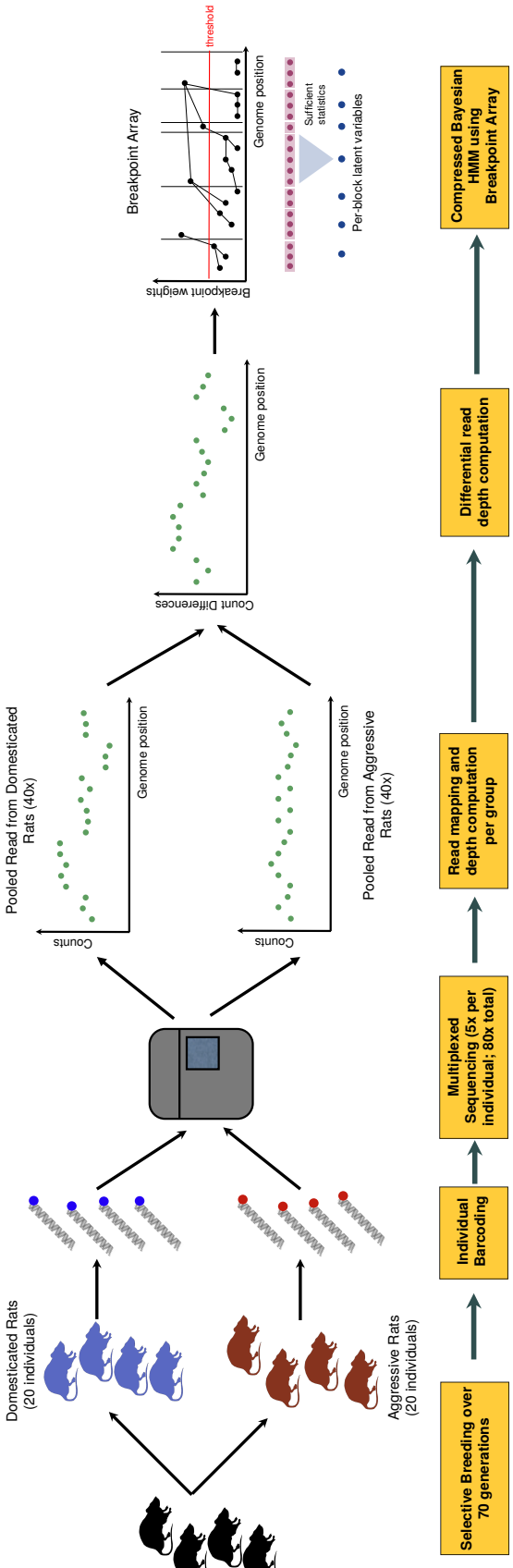


Figure 6.1: Experimental setup for the domestication experiment in rats.

selected for a reduced fear response towards an approaching human hand. 20 samples were obtained from the aggressive line, which has been selected for an increase in fearful and aggressive behavior towards an approaching human hand. DNA extraction was carried out at the Institute of Cytology and Genetics, the Siberian Branch of the Russian Academy of Sciences, Novosibirsk and at the Max Planck Institute for Evolutionary Anthropology (MPI-EVA), Germany.

For all samples, sequencing libraries were generated consisting of 125 bp double-indexed paired-end reads. Samples were pooled into a single library in order to avoid any batch effects during sequencing. Sequencing was performed on a combination of the Illumina Genome Analyzer II and High-Seq platforms. Library preparation and sequencing was carried out at the MPI-EVA. The rats have a mean coverage of ∼4X per individual, meaning that each genomic position is expected to be covered by 4 reads. Base calling was done using freeIbis (Renaud, Kircher, et al. 2013). Adapters were removed and potentially chimeric sequences flagged using the software leeHom using default parameters (Renaud, Stenzel & Kelso 2014). Reads were demultiplexed using deML using default quality thresholds (Renaud, Stenzel, Maricic, et al. 2015). Reads were then mapped to the Rattus norvegicus reference assembly rno5, using the Burrows-Wheeler Aligner (BWA) with default parameters (Li & Durbin 2009). Duplicate read removal was performed with Picard (http://broadinstitute.github.io/picard/). Local indel realignment was performed using the Genome Analysis Toolkit (GATK) (McKenna et al. 2010; DePristo et al. 2011; Van der Auwera et al. 2013).

6.3 Data prepocessing

We used SAMtools to process the BAM files resulting from the read mapping procedure. Lowest mapping positions were recorded for each read, and their counts were accumulated. Start counts for the tame population were subtracted from their counterparts in the aggressive population, yielding 1,880,703,547 data points. Note that, due to the multiplexed sequencing, any additive bias cancels out during subtraction. We used lowest mapped positions per read instead of per-base coverage (*pileup* data) for two reasons. Hidden Markov Models assume conditional independence of the observed data points given the state sequence. However, using pileup creates statistical dependence between neighboring positions due to reads covering multiple bases. Secondly,

pileup data tends to produce smooth curves. Since the Haar wavelet only has one vanishing moment, the local curvature of the data tends to produce large coefficients which severely reduces the compression. Additionally, sufficient statistics of blocks that are being created are hard to interpret, and the noise violates the Gaussian assumption. Instead, using only one position for each read decorrelates neighboring data points.

Due to the low coverage and the integer nature of the counts, the data showed highly discrete noise, and hence the data was averaged over non-overlapping windows of 20 positions to approximate Gaussian noise, resulting in 94,035,178 input positions. We then ran HaMMLET with 8 CNV states and automatic priors, see Wiedenhoeft, Brugel & Schliep 2016c. An example plot from this study showing complex CNV can be found in Fig. 6.2.

6.4 Benchmarks

On a computer with Intel Xeon CPU E7-8890 v4 (2.20 GHz) and 1 TB RAM, running Ubuntu 14.04.5 LTS, full Bayesian inference with HaMMLET for 200 iterations with a burn-in of 1,800 for an 8-state-model required 3 min 41 s and 1.3 GB RAM. By comparison, the previously published version of HaMMLET took 1 h 5 min 27 s, using 40 GB RAM (cf. Figure 6.3).

For a broader evaluation, we have created 100 replicates of the data by splitting it into 2500 chunks of equal sizes, which we then permuted randomly. We measured the memory usage (maximum resident set size), running time as well as cache behavior (minor page faults), see Fig. 6.4. The smaller savings in runtime compared to the original data can be attributed to the fact that permutation of the data is likely to disrupt long, highly compressible sections of the data. Also, with smaller blocks created under optimal compression, the undercompression effects in the wavelet tree might be less pronounced, hence its lower average runtime compared to the original data.

While the RAM usage remains almost constant among replicates within each implementation, we noticed that runtime and cache behavior varied widely in the old, but not the new implementation. We attribute this to the fact that the old compression scheme is suboptimal, yielding smaller blocks and hence more randomized assignment to states, leading to slower mixing properties of the Gibbs sampler. The data wavelet tree data contains outliers not shown

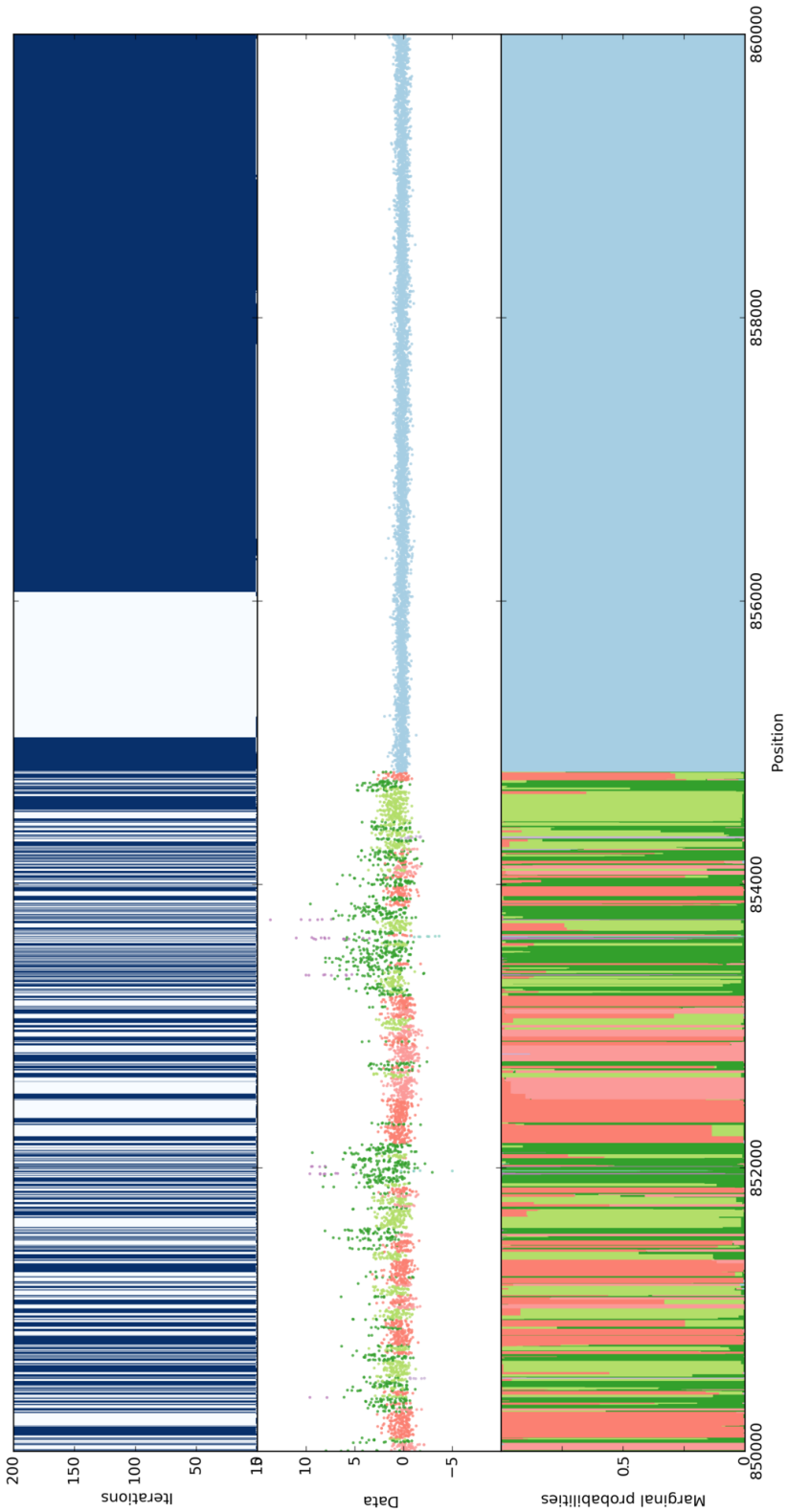


Figure 6.2: Example of CNV inference in multiplexed differential read count data using HaMMLET. The middle lane shows the differential read counts averaged over 20 position, and colored by their maximum marginal CNV state. The top lane shows the blocks created by the dynamic compression scheme. Block boundaries correspond to active values in the breakpoint array, and hence to the discontinuities in the Haar wavelet regression below a given threshold. As that threshold depends on the minimum emission variance in the HMM states at each iteration of the Gibbs sampler, the fact that the block structure does not change except for the first few iterations indicates that the FBG has run to convergence. Notice how the block structure automatically adapts to the local smoothness of the data. On the left, a high number of block boundaries indicates the presence of many discontinuities in the signal, allowing for fine-grained detection of CNV, whereas on the right-hand side, the scarcity of discontinuities indicates that the data is consistent with a model assuming constant signal and centered noise, and hence the absence of CNV. The bottom lane shows the marginal state probabilities for each segment, allowing for high-resolution inference of CNV. The data has not undergone corrections for amplification bias such as CG content; since all sequencing reads in multiplexed sequencing have the same amplification bias, taking the difference of their counts leads to data which does not exhibit long-range wave effects. The data compression factor over the entire data set is ∼ 630.

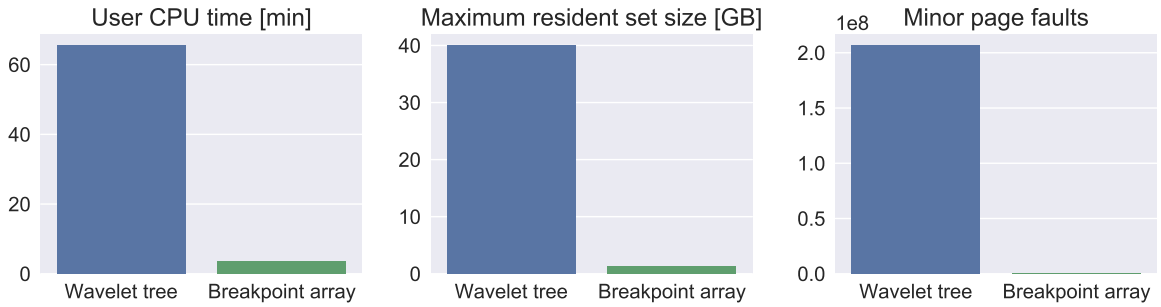


Figure 6.3: Comparison of benchmarks for running time, memory usage and cache behavior between the old and new versions of HaMMLET on the rat population WGS data set. The new approach yields a 17.8-fold speedup and 32.2-fold memory reduction. Notice that the number of minor page faults decreases by five orders of magnitude, indicating much better cache behavior due to the use of new data structures and an improved implementation. The number of major page faults is zero in both implementations.

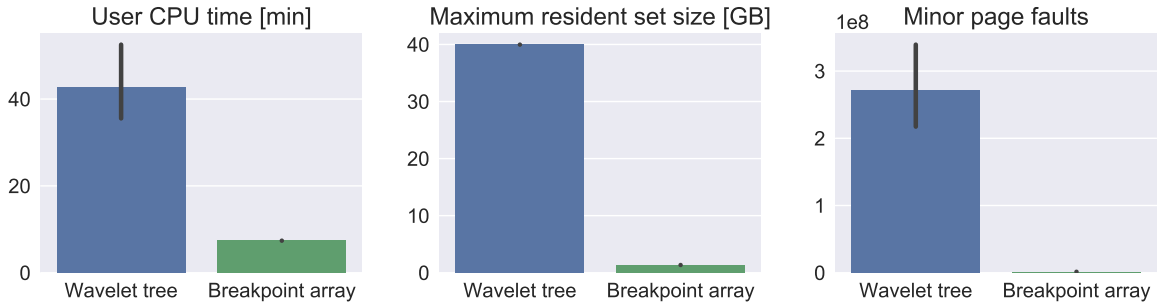


Figure 6.4: Comparison of benchmarks for running time, memory usage and cache behavior between the old and new versions of HaMMLET on 100 permutations of the rat population WGS data set. Notice the decrease in variance for both page faults and running time.

in the figure, most notably a runtime instance of 6.4 h, which is likely to result from sampling small emission variances due to short compression blocks.

6.5 Results

We consider all genomic segments with an absolute state mean ≥ 1 as containing putative structural variation segregating between the tame and aggressive rat lines. This results in 10,083,374 regions with a mean size of 407 base pairs. We identify all genes that are within or overlap these regions by ≥ 1 base pair using Ensembl's Variant Effect Predictor (McLaren et al. 2016). We find 1,036 genes with at least partial overlap with these regions.

To investigate the potential phenotypic consequences of these structural variants we performed a gene enrichment analysis using the software Webgestalt (Zhang, Kirov & Snoddy 2005; Wang, Duncan, et al. 2013). We tested for enrichment of gene ontology (GO term) categories for all

genes overlapping these structural variants using all genes in the rat genome as background. We consider as significantly enriched all pathways with p-value <0.05 after using the Benjamini and Hochberg procedure to correct for multiple hypothesis testing (Benjamini & Hochberg 1995). We identify many significantly enriched pathways, see Appendix A. We now briefly discuss some of these pathways and the genes within them and how they may inform us about the genetic changes underlying the phenotypic differences between these lines.

The most significantly enriched pathway is "Synapse assembly" (p-value = 0.0028), with five genes that are in putative structural variants segregating between the tame and aggressive rat lines. Some of these genes are associated with phenotypes that may be involved in the behavioral differences observed between the tame and aggressive rat lines. For example, one of the genes is the neuronal cadherin gene Cdh2. Missense mutations in this gene are associated with obsessive-compulsive behavior and Tourette disorder phenotypes in humans (Moya et al. 2013) and this gene has been associated with anxiety in mice (Donner et al. 2008). Another gene encodes the ephrin receptor Ephb1. The ephrin receptor-ligand system is involved in the regulation of several developmental processes in the nervous system. Notably, mice with null mutations for this gene exhibit neuronal loss in the substantia nigra and display spontaneous locomotor hyperactivity (Richards et al. 2007). This is interesting given that the tame and aggressive rats have differences in their activity in an open-field test (Albert et al. 2008).

We also observe multiple additional enriched pathways involved in neuronal development and function, e.g. "transmission of nerve impulse", "regulation of neurological system process", "dendrite morphogenesis". Therefore, we suspect that many of these segregating structural variants may have been targeted by selection and are contributing the phenotypic differences between these lines. Pending experimental validation, future study of the variants identified here may lead to insights into the domestication process.

Chapter 7

Conclusion

In this thesis, we have presented a novel approach for HMM-based Bayesian segmentation of large-scale data for the inference of genomic copy-number variants. Using a decision-theoretic framework, we have shown that for homoscedastic Gaussian emissions the discontinuities in a Haar wavelet regression based on the universal threshold capture strong changes in the posterior state distribution. This allows for a compression of the data into blocks of sufficient statistics within which the data is likely to be emitted from the same latent state. We derived the bias incurred in the forward-variables by ignoring between-state transitions for general HMM, generalizing earlier results based on the weak path assumption. The tendency of this forward bias to favor the state with the highest marginal likelihood within each block made us conjecture that Forward-Backward Gibbs sampling would result in a maximum posterior margins (MPM) segmentation of the data.

For heteroscedastic Gaussian HMM, we developed a Forward-Backward Gibbs sampling scheme based on dynamic Haar wavelet compression, in which the universal threshold is derived from the smallest emission variance sampled in each iteration. We developed the *wavelet tree* as a data structure to facilitate this sampling, and used extensive simulation studies to demonstrate the performance of our method. We show that compression greatly improves convergence, in terms of the number of sampling iterations, towards the true latent state sequence compared to uncompressed sampling, thus providing experimental support for our MPM segmentation conjecture. This faster convergence is achieved while also improving overall running time by two orders of magnitude. We also demonstrated the performance on biological gold-standard data

sets, improving upon the state of the art methods while running substantially faster.

We then showed that the wavelet tree yields a suboptimal compression, and derived a new compression scheme. We replaced the wavelet tree by more efficient and less memory-consuming data structures called *breakpoint array* and *integral array* to facilitate fast dynamic queries of the block sufficient statistics under arbitrary thresholds, without recomputing the Haar transform at every iteration. For an efficient constructor of these data structures, we developed a linear-time, in-place algorithm called *maxlet transform* to compute the maximum absolute detail coefficients, as well as an equivalent algorithm for multivariate data which incurs only logarithmic overhead in a streaming setting. We also developed a linear-time, in-place algorithm called the *Haar boundary transform* to compute the maximum of all detail coefficients affecting the discontinuities in the Haar wavelet regression at each position given arbitrary thresholds. Furthermore, we provided an efficient, queue-based data structure to store and update posterior state marginals during Gibbs sampling. Beyond a proof of concept, we have released our implementation in a software called HaMMLET under an open-source license at https://github.com/wiedenhoeft/HaMMLET. We have demonstrated that large-scale HMM inference can be performed on consumer hardware within a few minutes. We have demonstrated that plausible CNV candidates can be found using large-scale whole-genome sequencing data, and derived new scientific hypotheses about the role of CNV in the domestication syndrome. We are confident that HaMMLET is widely applicable to experimental data from current and future CNV technologies.

While we conjecture that HaMMLET achieves an approximation of MPM segmentation based on theoretical considerations and experimental results, the approximation of true posterior state margins remains an open challenge. Corrections for forward bias are not straightforward under constantly changing emission parameters during Gibbs sampling. Such a correction would also lead to the paradoxical situation that compression assumes exchangeability within each block, yet one of the fundamental properties of HMM is the ability to capture non-exchangeability of consecutive observations. We have ignored this question in the scope of this thesis, since real-life applications of CNV detection are often consistent with the weak path assumption. For more general applications, careful consideration should be given to those questions.

Appendix A

Significant pathways in rat populations

In this appendix, we report the genes affected by CNV segregating between tame and aggressive rat populations, as inferred by HaMMLET. Tables represent GO term enrichment for the "biological process" sub-root of the gene ontology. Tables are sorted according to the most significant p-value corrected for multiple testing. The columns are: Gene symbol, Description, Entrez Gene ID, and the Ensembl ID with the leading "ENSRNOG" (Ensembl prefix for *Rattus norvegicus*) as well as the padding zeros removed.

GO:0007416: synapse assembly ($p = 0.0028$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Nrg1	neuregulin 1	112400	10392
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0010646: regulation of cell communication ($p = 0.0064$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Myo9b	myosin IXb	25486	16256
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Plau	plasminogen activator, urokinase	25619	10516
Sall1	sal-like 1 (Drosophila)	307740	–
Cyth1	cytohesin 1	116691	43381
Epha4	Eph receptor A4	316539	13213
Ppp3cb	protein phosphatase 3, catalytic subunit, beta isozyme	24675	7757
Scoc	short coiled-coil protein	364981	3853
Egfr	epidermal growth factor receptor	24329	4332
Nrg1	neuregulin 1	112400	10392
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Rheb	Ras homolog enriched in brain	26954	–
Uaca	uveal autoantigen with coiled-coil domains and ankyrin repeats	315732	–
GO:0051674: localization of cell ($p = 0.0064$)			
Cdh2	cadherin 2	83501	15602
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Plau	plasminogen activator, urokinase	25619	10516
Dcdc2	doublecortin domain containing 2	291130	17511
Vcl	vinculin	305679	10765

GO:0030010: establishment of cell polarity ($p = 0.0064$)			
Sdccag8	serologically defined colon cancer antigen 8	305002	4181
Ephb1	Eph receptor B1	24338	7865
Cyth1	cytohesin 1	116691	43381
Myo9b	myosin IXb	25486	16256
GO:0050808: synapse organization ($p = 0.0064$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Nrg1	neuregulin 1	112400	10392
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0051969: regulation of transmission of nerve impulse ($p = 0.0064$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Egfr	epidermal growth factor receptor	24329	4332
Cdh2	cadherin 2	83501	15602
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Rheb	Ras homolog enriched in brain	26954	–
GO:0016477: cell migration ($p = 0.0064$)			
Cdh2	cadherin 2	83501	15602
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Plau	plasminogen activator, urokinase	25619	10516
Dcdc2	doublecortin domain containing 2	291130	17511
Vcl	vinculin	305679	10765

GO:0031644: regulation of neurological system process ($p = 0.0064$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Egfr	epidermal growth factor receptor	24329	4332
Cdh2	cadherin 2	83501	15602
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Rheb	Ras homolog enriched in brain	26954	–
GO:0046777: protein autophosphorylation ($p = 0.0064$)			
Egfr	epidermal growth factor receptor	24329	4332
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0048870: cell motility ($p = 0.0064$)			
Cdh2	cadherin 2	83501	15602
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Plau	plasminogen activator, urokinase	25619	10516
Dcdc2	doublecortin domain containing 2	291130	17511
Vcl	vinculin	305679	10765
GO:0021955: central nervous system neuron axonogenesis ($p = 0.0064$)			
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0050804: regulation of synaptic transmission ($p = 0.0068$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Egfr	epidermal growth factor receptor	24329	4332
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Rheb	Ras homolog enriched in brain	26954	–

GO:0048813: dendrite morphogenesis ($p = 0.0068$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Dcdc2	doublecortin domain containing 2	291130	17511
GO:0019226: transmission of nerve impulse ($p = 0.0068$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Egfr	epidermal growth factor receptor	24329	4332
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Rheb	Ras homolog enriched in brain	26954	–
GO:0055083: monovalent inorganic anion homeostasis ($p = 0.0068$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0055064: chloride ion homeostasis ($p = 0.0068$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0030644: cellular chloride ion homeostasis ($p = 0.0068$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0030320: cellular monovalent inorganic anion homeostasis ($p = 0.0068$)			
Stab2	stabilin 2	282580	–
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Cyth1	cytohesin 1	116691	43381
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Plau	plasminogen activator, urokinase	25619	10516
Vcl	vinculin	305679	10765
Cdh19	cadherin 19, type 2	360835	29841
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916

GO:0007155: cell adhesion ($p = 0.0068$)			
Stab2	stabilin 2	282580	–
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Cyth1	cytohesin 1	116691	43381
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Plau	plasminogen activator, urokinase	25619	10516
Vcl	vinculin	305679	10765
Cdh19	cadherin 19, type 2	360835	29841
GO:0006928: cellular component movement ($p = 0.0068$)			
Cdh2	cadherin 2	83501	15602
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Elmo1	engulfment and cell motility 1	361251	18726
Egfr	epidermal growth factor receptor	24329	4332
Plau	plasminogen activator, urokinase	25619	10516
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Dcdc2	doublecortin domain containing 2	291130	17511
Vcl	vinculin	305679	10765
GO:0035637: multicellular organismal signaling ($p = 0.0089$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Egfr	epidermal growth factor receptor	24329	4332
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Rheb	Ras homolog enriched in brain	26954	–

GO:0007268: synaptic transmission ($p = 0.0089$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Ephb1	Eph receptor B1	24338	7865
Egfr	epidermal growth factor receptor	24329	4332
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Rheb	Ras homolog enriched in brain	26954	–
GO:0007267: cell-cell signaling ($p = 0.0089$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Ephb1	Eph receptor B1	24338	7865
Ppp3cb	protein phosphatase 3, catalytic subunit, beta isozyme	24675	7757
Egfr	epidermal growth factor receptor	24329	4332
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Rheb	Ras homolog enriched in brain	26954	–
Sall1	sal-like 1 (Drosophila)	307740	–
GO:0048858: cell projection morphogenesis ($p = 0.0105$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Dcdc2	doublecortin domain containing 2	291130	17511

GO:0000902: cell morphogenesis ($p = 0.0105$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Dcdc2	doublecortin domain containing 2	291130	17511
Sall1	sal-like 1 (Drosophila)	307740	–
GO:0007163: establishment or maintenance of cell polarity ($p = 0.0105$)			
Sdccag8	serologically defined colon cancer antigen 8	305002	4181
Ephb1	Eph receptor B1	24338	7865
Cyth1	cytohesin 1	116691	43381
Myo9b	myosin IXb	25486	16256
GO:0032990: cell part morphogenesis ($p = 0.0114$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Dcdc2	doublecortin domain containing 2	291130	17511
GO:0040011: locomotion ($p = 0.0114$)			
Cdh2	cadherin 2	83501	15602
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Plau	plasminogen activator, urokinase	25619	10516
Dcdc2	doublecortin domain containing 2	291130	17511
Vcl	vinculin	305679	10765

GO:0050770: regulation of axonogenesis ($p = 0.0114$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Epha4	Eph receptor A4	316539	13213
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0023051: regulation of signaling ($p = 0.0114$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Myo9b	myosin IXb	25486	16256
Gria4	glutamate receptor, ionotropic, AMPA 4	29629	6957
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Plau	plasminogen activator, urokinase	25619	10516
Sall1	sal-like 1 (Drosophila)	307740	–
Cyth1	cytohesin 1	116691	43381
Epha4	Eph receptor A4	316539	13213
Ppp3cb	protein phosphatase 3, catalytic subunit, beta isozyme	24675	7757
Egfr	epidermal growth factor receptor	24329	4332
Nrg1	neuregulin 1	112400	10392
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Rheb	Ras homolog enriched in brain	26954	–
Uaca	uveal autoantigen with coiled-coil domains and ankyrin repeats	315732	–
GO:0048812: neuron projection morphogenesis ($p = 0.0129$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Egfr	epidermal growth factor receptor	24329	4332
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Dcdc2	doublecortin domain containing 2	291130	17511

GO:0051179: localization ($p = 0.0129$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Ephb1	Eph receptor B1	24338	7865
Sec24c	SEC24 family, member C (S. cerevisiae)	685144	9042
Brca2	breast cancer 2	360254	1111
Myo9b	myosin IXb	25486	16256
Elmo1	engulfment and cell motility 1	361251	18726
Camk2g	calcium/calmodulin-dependent protein kinase II gamma	171140	9783
Plau	plasminogen activator, urokinase	25619	10516
Dcdc2	doublecortin domain containing 2	291130	17511
Vcl	vinculin	305679	10765
Tbc1d1	TBC1 (tre-2/USP6, BUB2, cdc16) domain family, member 1	360937	2180
Kcnt2	potassium channel, subfamily T, member 2	304827	13312
Stab2	stabilin 2	282580	–
Itln1	intelectin 1 (galactofuranose binding)	498284	4678
Ppp3cb	protein phosphatase 3, catalytic subunit, beta isozyme	24675	7757
Epha4	Eph receptor A4	316539	13213
Cyth1	cytohesin 1	116691	43381
Ccdc91	coiled-coil domain containing 91	312863	–
Egfr	epidermal growth factor receptor	24329	4332
Nrg1	neuregulin 1	112400	10392
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Uaca	uveal autoantigen with coiled-coil domains and ankyrin repeats	315732	–
GO:0007265: Ras protein signal transduction ($p = 0.0142$)			
Elmo1	engulfment and cell motility 1	361251	18726
Cdh2	cadherin 2	83501	15602
Epha4	Eph receptor A4	316539	13213
Nrg1	neuregulin 1	112400	10392
Cyth1	cytohesin 1	116691	43381
Myo9b	myosin IXb	25486	16256

GO:0032989: cellular component morphogenesis ($p = 0.0142$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Myo9b	myosin IXb	25486	16256
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Dcdc2	doublecortin domain containing 2	291130	17511
Sall1	sal-like 1 (Drosophila)	307740	–
GO:0030002: cellular anion homeostasis ($p = 0.0177$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0018212: peptidyl-tyrosine modification ($p = 0.0187$)			
Egfr	epidermal growth factor receptor	24329	4332
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Epha4	Eph receptor A4	316539	13213
Nrg1	neuregulin 1	112400	10392
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0018108: peptidyl-tyrosine phosphorylation ($p = 0.0187$)			
Egfr	epidermal growth factor receptor	24329	4332
Ddr2	discoidin domain receptor tyrosine kinase 2	685781	2881
Epha4	Eph receptor A4	316539	13213
Nrg1	neuregulin 1	112400	10392
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
GO:0009894: regulation of catabolic process ($p = 0.0195$)			
Egfr	epidermal growth factor receptor	24329	4332
Epha4	Eph receptor A4	316539	13213
Nrg1	neuregulin 1	112400	10392
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Myo9b	myosin IXb	25486	16256
Uaca	uveal autoantigen with coiled-coil domains and ankyrin repeats	315732	–
Scoc	short coiled-coil protein	364981	3853

GO:0030182: neuron differentiation ($p = 0.0195$)			
Cacna1a	calcium channel, voltage-dependent, P/Q type, alpha 1A subunit	25398	2559
Cdh2	cadherin 2	83501	15602
Ephb1	Eph receptor B1	24338	7865
Epha4	Eph receptor A4	316539	13213
Egfr	epidermal growth factor receptor	24329	4332
Ptk2	PTK2 protein tyrosine kinase 2	25614	7916
Nrg1	neuregulin 1	112400	10392
Cdk5rap1	CDK5 regulatory subunit associated protein 1	252827	15696
Dcdc2	doublecortin domain containing 2	291130	17511
Sall1	sal-like 1 (Drosophila)	307740	–

Acknowledgement of Previous Publications

Bravo, Gustavo A., Alexandre Antonelli, Christine D. Bacon, Krzysztof Bartoszek, Mozes P. K. Blom, Stella Huynh, Graham Jones, L. Lacey Knowles, Sangeet Lamichhaney, Thomas Marcussen, Hélène Morlon, Luay K. Nakhleh, Bengt Oxelman, Bernard Pfeil, Alexander Schliep, Niklas Wahlberg, Fernanda P. Werneck, John Wiedenhoeft, Sandi Willows-Munro & Scott V. Edwards (Jan. 2018). "Embracing heterogeneity: building the Tree of Life and the future of phylogenomics". In: *PeerJ*. Preprint. doi: 10.7287/peerj.preprints.26449v1. url: https://peerj.com/preprints/26449/.

Günzel, Dorothee, Silke S. Zakrzewski, Thomas Schmid, Maria Pangalos, John Wiedenhoeft, Corinna Blasse, Christopher Ozboda & Susanne M. Krug (2012). "From TER to trans- and paracellular resistance: lessons from impedance spectroscopy". In: *Annals of the New York Academy of Sciences* 1257.1, pp. 142–151. issn: 1749-6632. doi: 10.1111/j.1749-6632.2012.06540.x. PMID: 22671600. url: http://onlinelibrary.wiley.com/doi/10.1111/j.1749-6632.2012.06540.x/full.

Homilius, Max, John Wiedenhoeft, Sebastian Thieme, Christoph Standfuss, Ivan Kel & Roland Krause (2011). "Cocos: Constructing multi-domain protein phylogenies". In: *PLoS Currents: Tree of Life*. doi: 10.1371/currents.RRN1240. PMID: PMC3110499. url: http://currents.plos.org/treeoflife/article/cocos-constructing-multi-domain-protein-phylogenies/.

Mahmud, Md Pavel, John Wiedenhoeft & Alexander Schliep (2012). "Indel-tolerant read mapping with trinucleotide frequencies using cache-oblivious *kd*-trees". In: *Bioinformatics*. doi: 10.1093/bioinformatics/bts380. PMID: PMC3436807. url: http://bioinformatics.oxfordjournals.org/content/28/18/i325.

Wiedenhoeft, John, Eric Brugel & Alexander Schliep (2016a). "Fast Bayesian Inference of Copy Number Variants using Hidden Markov Models with Wavelet Compression". In: *PLOS Computational Biology* 5 (12). doi: 10.1371/journal.pcbi.1004871. url: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1004871.

– (2016b). "Fast Bayesian Inference of Copy Number Variants using Hidden Markov Models with Wavelet Compression". In: *Research in Computational Molecular Biology: 20th Annual Conference, RECOMB 2017*. This talk was a parallel submission to the PLOS paper of the same title. isbn: 978-3-319-31957-5.

Wiedenhoeft, John, Alex Cagan, Rimma Kozhemjakina, Rimma Gulevich & Alexander Schliep (2017). "Locating CNV candidates in WGS data using wavelet-compressed Bayesian HMM". In: *ISMB/ECCB 2017*. Conference poster. url: https://f1000research.com/posters/6-1418.

Wiedenhoeft, John, Alex Cagan, Rimma Kozhemyakina, Rimma Gulevich & Alexander Schliep (2018). "Bayesian localization of CNV candidates in WGS data within minutes". In: *Submitted to BMC Algorithms for Molecular Biology*.

Wiedenhoeft, John, Roland Krause & Oliver Eulenstein (2010). "Inferring Evolutionary Scenarios for Protein Domain Compositions". In: *6th International Symposium on Bioinformatics Research and*

Applications. Ed. by Mark Borodovsky, Johann Peter Gogarten, Teresa M. Przytycka & Sanguthevar Rajasekaran. Vol. 6053. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 179–190. isbn: 978-3-642-13077-9. doi: 10.1007/978-3-642-13078-6_21. url: http://www.springerlink.com/content/5260h5m115v27ng8/.

Wiedenhoeft, John, Roland Krause & Oliver Eulenstein (2011). "The Plexus Model for the Inference of Ancestral Multi-Domain Proteins". In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 8.4, pp. 890–901. issn: 1557-9964. doi: 10.1109/TCBB.2011.22. url: http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=5708135.

Wiedenhoeft, John & Alexander Schliep (2017). "Using HaMMLET for Bayesian Segmentation of WGS Read-Depth Data". In: *Copy Number Variants: Methods and Protocols*. Ed. by Derek Bickhart. Methods in Molecular Biology 1833. Springer, pp. 83–93. doi: 10.1007/978-1-4939-8666-8_6. url: https://link.springer.com/protocol/10.1007%2F978-1-4939-8666-8_6.

– (2018). "Decision-theoretic foundation of Bayesian HMM inference under Haar wavelet compression". In: *Submitted to NIPS 2018*.

Bibliography

Abel, Haley J. & Eric J. Duncavage (2013). "Detection of structural DNA variation from next generation sequencing data: a review of informatic approaches". In: *Cancer Genetics* 206.12, pp. 432–440. issn: 22107762. doi: 10.1016/j.cancergen.2013.11.002. PMID: 24405614. url: http://linkinghub.elsevier.com/retrieve/pii/S2210776213001579.

Albert, Frank W., Olesya Shchepina, Christine Winter, Holger Römpler, Daniel Teupser, Rupert Palme, Uta Ceglarek, Jürgen Kratzsch, Reinhard Sohr, Lyudmila N. Trut, Joachim Thiery, Rudolf Morgenstern, Irina Z. Plyusnina, Torsten Schöneberg & Svante Pääbo (2008). "Phenotypic differences in behavior, physiology and neurochemistry between rats selected for tameness and for defensive aggression towards humans". In: *Hormones and Behavior* 53.3, pp. 413–421. issn: 0018506X. doi: 10.1016/j.yhbeh.2007.11.010. url: http://linkinghub.elsevier.com/retrieve/pii/S0018506X07002814.

Alkan, Can, Bradley P. Coe & Evan E. Eichler (2011). "Genome structural variation discovery and genotyping". In: *Nature Reviews Genetics* 12.5, pp. 363–376. issn: 1471-0056. doi: 10.1038/nrg2958. url: http://www.nature.com/doifinder/10.1038/nrg2958.

Antoniadis, Anestis & Georges Oppenheim (1995). *Wavelets and Statistics*. Ed. by Anestis Antoniadis & Georges Oppenheim. Vol. 103. Lecture Notes in Statistics. New York, NY: Springer New York. isbn: 978-0-387-94564-4. doi: 10.1007/978-1-4612-2544-7. url: http://link.springer.com/10.1007/978-1-4612-2544-7.

Autio, Reija, Sampsa Hautaniemi, Päivikki Kauraniemi, Olli Yli-Harja, Jaakko Astola, Maija Wolf & Anne Kallioniemi (2003). "CGH-Plotter: MATLAB toolbox for CGH-data analysis". In: *Bioinformatics* 19.13, pp. 1714–1715. issn: 1367-4803. doi: 10.1093/bioinformatics/btg230. url: http://bioinformatics.oxfordjournals.org/content/19/13/1714.abstract.

Axelsson, Erik, Abhirami Ratnakumar, Maja-Louise Arendt, Khurram Maqbool, Matthew T. Webster, Michele Perloski, Olof Liberg, Jon M. Arnemo, Åke Hedhammar & Kerstin Lindblad-Toh (2013). "The genomic signature of dog domestication reveals adaptation to a starch-rich diet". In: *Nature* 495.7441, pp. 360–364. issn: 0028-0836. doi: 10.1038/nature11837. url: http://www.nature.com/doifinder/10.1038/nature11837.

Barry, D. A., J.-Y. Parlange, L. Li, H. Prommer, C. J. Cunningham & F. Stagnitti (2000). "Analytical approximations for real values of the Lambert W-function". In: *Mathematics and Computers in Simulation* 53.1-2, pp. 95–103. issn: 03784754. doi: 10.1016/S0378-4754(00)00172-5. url: http://www.sciencedirect.com/science/article/pii/S0378475400001725.

Bassett, Robert & Julio Deride (2018). "Maximum a posteriori estimators as a limit of Bayes estimators". In: *Mathematical Programming*, pp. 1–16. issn: 0025-5610. doi: 10.1007/s10107-018-1241-0. url: http://link.springer.com/10.1007/s10107-018-1241-0.

Baum, Leonard E. & Ted Petrie (1966). "Statistical Inference for Probabilistic Functions of Finite State Markov Chains". In: *The Annals of Mathematical Statistics* 37.6, pp. 1554–1563. issn: 0003-4851. doi: 10.1214/aoms/1177699147. url: http://www.jstor.org/stable/2238772.

Belyaev, Dmitri K. (1969). "Domestication of animals". In: *Science Journal* 5.1, pp. 47–52.

Ben-Yaacov, Erez & Yonina C. Eldar (2008). "A fast and flexible method for the segmentation of aCGH data." In: *Bioinformatics* 24.16, pp. i139–45. issn: 1367-4811. doi: 10.1093/bioinformatics/btn272. PMID: 18689815. url: http://bioinformatics.oxfordjournals.org/content/24/16/i139.long.

Benjamini, Yoav & Yosef Hochberg (1995). "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 57, pp. 289–300. doi: 10.2307/2346101. url: https://www.jstor.org/stable/2346101.

Berman, Simeon M. (1992). *Sojourns and extremes of stochastic processes*. CRC Press. isbn: 978-0-534-13932-2.

Bilmes, Jeff a. (1998). "A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models". In: *International Computer Science Institute* 4.510, p. 126. url: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.613.

Boveri, Theodor (1914). *Zur Frage der Entstehung maligner Tumoren*. Jena: Gustav Fischer.

Burdall, Sarah, Andrew Hanby, Mark Lansdown & Valerie Speirs (2003). "Breast cancer cell lines: friend or foe?" In: *Breast Cancer Research* 5.2, pp. 89–95. issn: 1465-5411. doi: 10.1186/bcr577. url: http://breast-cancer-research.com/content/5/2/89.

Cahan, Patrick, Laura E. Godfrey, Peggy S. Eis, Todd A. Richmond, Rebecca R. Selzer, Michael Brent, Howard L. McLeod, Timothy J. Ley & Timothy A. Graubert (2008). "wuHMM: a robust algorithm to detect DNA copy number variation using long oligonucleotide microarray data." In: *Nucleic Acids Research* 36.7, e41. issn: 1362-4962. doi: 10.1093/nar/gkn110. PMID: 18334530. url: http://nar.oxfordjournals.org/content/36/7/e41.abstract.

Chib, Siddhartha (1996). "Calculating posterior distributions and modal estimates in Markov mixture models". In: *Journal of Econometrics* 75.1, pp. 79–97. url: http://www.sciencedirect.com/science/article/pii/0304407695017704.

Chung, Brian Hon-Yin, Victoria Qinchen Tao & Winnie Wan-Yee Tso (2014). "Copy number variation and autism: New insights and clinical implications". In: *Journal of the Formosan Medical Association* 113.7, pp. 400–408. doi: 10.1016/j.jfma.2013.01.005.

Cleveland, William S. (1979). "Robust Locally Weighted Regression and Smoothing Scatterplots". en. In: *Journal of the American Statistical Association* 74.368. doi: 10.1080/01621459.1979.10481038. url: http://www.tandfonline.com/doi/abs/10.1080/01621459.1979.10481038.

Conrad, Donald F., T. Daniel Andrews, Nigel P. Carter, Matthew E. Hurles & Jonathan K. Pritchard (2006). "A high-resolution survey of deletion polymorphism in the human genome". In: *Nature Genetics* 38.1, pp. 75–81. issn: 1061-4036. doi: 10.1038/ng1697. url: http://www.nature.com/doifinder/10.1038/ng1697.

Cook Jr, Edwin H. & Stephen W. Scherer (2008). "Copy-number variations associated with neuropsychiatric conditions". In: *Nature* 455.7215, pp. 919–923. issn: 0028-0836. doi: 10.1038/nature07458. PMID: 18923514. url: http://www.nature.com/articles/nature07458.

Darmois, G. (1935). "Sur les lois de probabilité a estimation exhaustive". In: *Comptes Rendus de l'Académie des Sciences* 260.1265, p. 85.

Darwin, Charles (1868). *The Variation in Animals and Plants under Domestication*. London: John Murray.

De Vries, Bert B. A., Rolph Pfundt, Martijn Leisink, David A. Koolen, Lisenka E. L. M. Vissers, Irene M. Janssen, Simon van Reijmersdal, Willy M. Nillesen, Erik H. L. P. G. Huys, Nicole de Leeuw, Dominique Smeets, Erik A. Sistermans, Ton Feuth, Conny M. A. van Ravenswaaij-Arts, Ad Geurts van Kessel, Eric F. P. M. Schoenmakers, Han G. Brunner & Joris A. Veltman (2005). "Diagnostic Genome Profiling in Mental Retardation". In: *The American Journal of Human Genetics* 77.4, pp. 606–616. issn: 00029297. doi: 10.1086/491719. PMID: 16175506. url: http://www.sciencedirect.com/science/article/pii/S0002929707610088.

DePristo, Mark A., Eric Banks, Ryan Poplin, Kiran V. Garimella, Jared R. Maguire, Christopher Hartl, Anthony A. Philippakis, Guillermo del Angel, Manuel A. Rivas, Matt Hanna, Aaron McKenna, Tim J. Fennell, Andrew M. Kernytsky, Andrey Y. Sivachenko, Kristian Cibulskis, Stacey B. Gabriel, David Altshuler & Mark J. Daly (2011). "A framework for variation discovery and genotyping using next-generation DNA sequencing data". In: *Nature Genetics* 43.5, pp. 491–498. issn: 1061-4036. doi: 10.1038/ng.806. url: http://www.nature.com/doifinder/10.1038/ng.806.

Diaconis, Persi & Donald Ylvisaker (1979). "Conjugate Priors for Exponential Families". In: *The Annals of Statistics* 7.2, pp. 269–281. issn: 0090-5364. doi: 10.1214/aos/1176344611. url: http://projecteuclid.org/euclid.aos/1176344611.

Diamond, Jared M. (1998). *Guns, germs and steel: a short history of everybody for the last 13,000 years*. Random House.

Donner, Jonas, Sami Pirkola, Kaisa Silander, Laura Kananen, Joseph D. Terwilliger, Jouko Lönnqvist, Leena Peltonen & Iiris Hovatta (2008). "An association analysis of murine anxiety genes in humans implicates novel candidate genes for anxiety disorders." In: *Biological psychiatry* 64.8, pp. 672–80. issn: 1873-2402. doi: 10.1016/j.biopsych.2008.06.002. PMID: 18639233. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2682432.

Donoho, David L. & Iain M. Johnstone (1994). "Ideal spatial adaptation by wavelet shrinkage". In: *Biometrika* 81.3, pp. 425–455. issn: 0006-3444. doi: 10.1093/biomet/81.3.425. PMID: 18439138. url: http://biomet.oxfordjournals.org/content/81/3/425.

– (1995). "Adapting to Unknown Smoothness via Wavelet Shrinkage". In: *Journal Of The American Statistical Association* 90.432, pp. 1200–1224. issn: 01621459. doi: 10.2307/2291512. url: http://www.jstor.org/stable/2291512.

Donoho, David L., Iain M. Johnstone, Gerard Kerkyacharian & Dominique Picard (1995). "Wavelet Shrinkage: Asymptopia?" In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 57.2, pp. 301–369. url: http://www.jstor.org/stable/2345967.

Duan, Junbo, Ji-Gang Zhang, Hong-Wen Deng, Yu-Ping Wang & J. Ruan (2013). "Comparative studies of copy number variation detection methods for next-generation sequencing technologies." In: *PloS ONE* 8.3. Ed. by Nicolas Salamin, e59128. issn: 1932-6203. doi: 10.1371/journal.pone.0059128. PMID: 23527109. url: http://dx.plos.org/10.1371/journal.pone.0059128.

Edgren, Henrik, Astrid Murumagi, Sara Kangaspeska, Daniel Nicorici, Vesa Hongisto, Kristine Kleivi, Inga H. Rye, Sandra Nyberg, Maija Wolf, Anne-Lise Borresen-Dale & Olli Kallioniemi (2011). "Identification of fusion genes in breast cancer by paired-end RNA-sequencing". In: *Genome Biology* 12.1, R6. issn: 1465-6906. doi: 10.1186/gb-2011-12-1-r6. PMID: 21247443. url: http://genomebiology.biomedcentral.com/articles/10.1186/gb-2011-12-1-r6.

Eilers, Paul H. C. & Renée X. de Menezes (2005). "Quantile smoothing of array CGH data." In: *Bioinformatics* 21.7, pp. 1146–53. issn: 1367-4803. doi: 10.1093/bioinformatics/bti148. PMID: 15572474. url: http://bioinformatics.oxfordjournals.org/content/21/7/1146.full.

Fan, Jianqing, Peter Hall, Michael Martin & Prakash Patil (1993). "Adaptation to high spatial inhomogeneity based on wavelets and on local linear smoothing". In: *Institute of Statistics Mimeo Series* 2307.

Fisher, R. A. (1922). "On the Mathematical Foundations of Theoretical Statistics". In: *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character* 222, pp. 309–368. url: http://www.jstor.org/stable/91208.

Fog, Agner (2016). *Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs*. url: http://www.agner.org/optimize/instruction_tables.pdf.

Foijer, Floris, Viji M. Draviam & Peter K. Sorger (2008). "Studying chromosome instability in the mouse". In: *Biochimica et Biophysica Acta (BBA) - Reviews on Cancer* 1786.1, pp. 73–82. issn: 0304419X. doi: 10.1016/j.bbcan.2008.07.004. url: http://linkinghub.elsevier.com/retrieve/pii/S0304419X08000322.

Folstein, Susan E. & Beth Rosen-Sheidley (2001). "Genetics of austim: complex aetiology for a heterogeneous disorder". In: *Nature Reviews Genetics* 2.12, pp. 943–955. issn: 1471-0056. doi: 10.1038/35103559. url: http://www.nature.com/doifinder/10.1038/35103559.

Forney, G. D. (1973). "The Viterbi algorithm". In: *Proceedings of the IEEE* 61.3, pp. 268–278. issn: 0018-9219. doi: 10.1109/PROC.1973.9030. url: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1450960.

Fredman, David, Stefan J. White, Susanna Potter, Evan E. Eichler, Johan T. Den Dunnen & Anthony J. Brookes (2004). "Complex SNP-related sequence variation in segmental genome duplications". In: *Nature Genetics* 36.8, pp. 861–866. issn: 1061-4036. doi: 10.1038/ng1401. url: http://www.nature.com/doifinder/10.1038/ng1401.

Freedman, Matthew L., Alvaro N. A. Monteiro, Simon A. Gayther, Gerhard A. Coetzee, Angela Risch, Christoph Plass, Graham Casey, Mariella De Biasi, Chris Carlson, David Duggan, Michael James, Pengyuan Liu, Jay W. Tichelaar, Haris G. Vikis, Ming You & Ian G. Mills (2011). "Principles for the post-GWAS functional characterization of cancer risk loci". In: *Nature Genetics* 43.6, pp. 513–518. issn: 1061-4036. doi: 10.1038/ng.840. url: http://www.nature.com/doifinder/10.1038/ng.840.

Freeman, J. L. (2006). "Copy number variation: New insights in genome diversity". In: *Genome Research* 16.8, pp. 949–961. issn: 1088-9051. doi: 10.1101/gr.3677206. url: http://www.genome.org/cgi/doi/10.1101/gr.3677206.

Fridlyand, Jane, Antoine M. Snijders, Dan Pinkel, Donna G. Albertson & Ajay N. Jain (2004). "Hidden Markov models approach to the analysis of array CGH data". In: *Journal of Multivariate Analysis* 90.1, pp. 132–153. issn: 0047259X. doi: 10.1016/j.jmva.2004.02.008. url: http://www.sciencedirect.com/science/article/pii/S0047259X04000260.

Fröhling, Stefan & Hartmut Döhner (2008). "Chromosomal Abnormalities in Cancer". In: *New England Journal of Medicine* 359.7, pp. 722–734. issn: 0028-4793. doi: 10.1056/NEJMra0803109. url: http://www.nejm.org/doi/abs/10.1056/NEJMra0803109.

Garcia, M., A. Jemal, EM Ward, MM Center, Y. Hao, RL Siegel & MJ Thun (2007). *Global Cancer Facts and Figures 2007*. Atlanta, GA. url: http://www.cancer.org/research/cancerfactsfigures/globalcancerfactsfigures/global-cancer-facts-figures-2007.

Garraway, Levi A. & Eric S. Lander (2013). "Lessons from the Cancer Genome". In: *Cell* 153.1, pp. 17–37. issn: 00928674. doi: 10.1016/j.cell.2013.03.002. url: http://linkinghub.elsevier.com/retrieve/pii/S0092867413002882.

Ghahramani, Zoubin (2001). “An introduction to hidden Markov models and Bayesian networks”. In: *International Journal of Pattern Recognition and Artificial Intelligence* 15.01, pp. 9–42. doi: 10.1142/S0218001401000836. url: http://www.worldscientific.com/doi/abs/10.1142/S0218001401000836.

González, Juan R., Isaac Subirana, Geòrgia Escaramís, Solymar Peraza, Alejandro Cáceres, Xavier Estivill & Lluís Armengol (2009). “Accounting for uncertainty when assessing association between copy number and disease: a latent class model.” En. In: *BMC Bioinformatics* 10.1, p. 172. issn: 1471-2105. doi: 10.1186/1471-2105-10-172. PMID: 19500389. url: http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-10-172.

Guha, Subharup, Yi Li & Donna Neuberg (2006). *Bayesian Hidden Markov Modeling of Array CGH Data*. Tech. rep. Harvard University. url: http://biostats.bepress.com/harvardbiostat/paper24.

Haar, Alfréd (1910). “Zur Theorie der orthogonalen Funktionensysteme”. In: *Mathematische Annalen* 69.3, pp. 331–371. issn: 0025-5831. doi: 10.1007/BF01456326. url: http://link.springer.com/10.1007/BF01456326.

Halmos, Paul R. & L. J. Savage (1949). “Application of the Radon-Nikodym Theorem to the Theory of Sufficient Statistics”. In: *The Annals of Mathematical Statistics* 20.2, pp. 225–241. issn: 0003-4851. doi: 10.1214/aoms/1177730032. url: http://projecteuclid.org/euclid.aoms/1177730032.

Hastings, P. J., James R. Lupski, Susan M. Rosenberg & Grzegorz Ira (2009). “Mechanisms of change in gene copy number.” In: *Nature Reviews Genetics* 10.8, pp. 551–564. issn: 1471-0064. doi: 10.1038/nrg2593. PMID: 19597530. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2864001.

Hehir-Kwa, Jayne Y., Rolph Pfundt & Joris A. Veltman (2015). “Exome sequencing and whole genome sequencing for the detection of copy number variation”. In: *Expert Review of Molecular Diagnostics* 15.8, pp. 1023–1032. issn: 1473-7159. doi: 10.1586/14737159.2015.1053467. PMID: 26088785. url: http://www.tandfonline.com/doi/full/10.1586/14737159.2015.1053467.

Higham, Nicholas J. (1993). “The Accuracy of Floating Point Summation”. In: *SIAM Journal on Scientific Computing* 14.4, pp. 783–799. issn: 1064-8275. doi: 10.1137/0914050. url: http://epubs.siam.org/doi/10.1137/0914050.

Hodgson, Graeme, Jeffrey H. Hager, Stas Volik, Sujatmi Hariono, Meredith Wernick, Dan Moore, Donna G. Albertson, Daniel Pinkel, Colin Collins, Douglas Hanahan & Joe W. Gray (2001). “Genome scanning with array CGH delineates regional alterations in mouse islet carcinomas”. In: *Nature Genetics* 29.4, pp. 459–464. issn: 1061-4036. doi: 10.1038/ng771. PMID: 11694878. url: https://www.nature.com/articles/ng771.

Holland, Andrew J. & Don W. Cleveland (2009). “Boveri revisited: chromosomal instability, aneuploidy and tumorigenesis”. In: *Nature Reviews Molecular Cell Biology* 10.7, pp. 478–487. issn: 1471-0072. doi: 10.1038/nrm2718. url: http://www.nature.com/doifinder/10.1038/nrm2718.

Holliday, Deborah L. & Valerie Speirs (2011). “Choosing the right cell line for breast cancer research.” In: *Breast Cancer Research* 13.4, p. 215. issn: 1465-542X. doi: 10.1186/bcr2889. PMID: 21884641. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3236329.

Hsu, Li, Steven G. Self, Douglas Grove, Tim Randolph, Kai Wang, Jeffrey J. Delrow, Lenora Loo & Peggy Porter (2005). “Denoising array-based comparative genomic hybridization data using wavelets.” In: *Biostatistics (Oxford, England)* 6.2, pp. 211–26. issn: 1465-4644. doi: 10.1093/biostatistics/kxi004. PMID: 15772101. url: http://biostatistics.oxfordjournals.org/content/6/2/211.

Huang, Heng, Nha Nguyen, Soontorn Oraintara & An Vo (2008). “Array CGH data modeling and smoothing in Stationary Wavelet Packet Transform domain.” In: *BMC Genomics* 9 Suppl 2, S17. issn:

1471-2164. DOI: 10.1186/1471-2164-9-S2-S17. PMID: 18831782. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2559881.

HUPÉ, Philippe, Nicolas STRANSKY, Jean-Paul THIERY, François RADVANYI & Emmanuel BARILLOT (2004). "Analysis of array CGH data: from signal ratio to gain and loss of DNA regions." In: *Bioinformatics* 20.18, pp. 3413–22. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/bth418. PMID: 15381628. URL: http://bioinformatics.oxfordjournals.org/content/20/18/3413.abstract.

HURLES, Matthew E., Emmanouil T. DERMITZAKIS & Chris TYLER-SMITH (2008). "The functional impact of structural variation in humans". In: *Trends in Genetics* 24.5, pp. 238–245. ISSN: 01689525. DOI: 10.1016/j.tig.2008.03.001. URL: http://linkinghub.elsevier.com/retrieve/pii/S0168952508000784.

IAFRATE, A. John, Lars FEUK, Miguel N. RIVERA, Marc L. LISTEWNIK, Patricia K. DONAHOE, Ying QI, Stephen W. SCHERER & Charles LEE (2004). "Detection of large-scale variation in the human genome". In: *Nature Genetics* 36.9, pp. 949–951. ISSN: 1061-4036. DOI: 10.1038/ng1416. PMID: 15286789. URL: http://www.nature.com/ng/journal/v36/n9/full/ng1416.html.

KADALAYIL, Latha, Sajjad RAFIQ, Matthew J. J. ROSE-ZERILLI, Reuben J. PENGELLY, Helen PARKER, David OSCIER, Jonathan C. STREFFORD, William J. TAPPER, Jane GIBSON, Sarah ENNIS & Andrew COLLINS (2015). "Exome sequence read depth methods for identifying copy number changes". In: *Briefings in Bioinformatics* 16.3, pp. 380–392. ISSN: 1467-5463. DOI: 10.1093/bib/bbu027. URL: http://bib.oxfordjournals.org/cgi/doi/10.1093/bib/bbu027.

KAHAN, W. (1965). "Further remarks on reducing truncation errors". In: *Communications of the ACM* 8.1, p. 40. ISSN: 00010782. DOI: 10.1145/363707.363723. URL: https://dl.acm.org/citation.cfm?doid=363707.363723.

KANNER, Leo (1943). "Autistic disturbances of affective contact". In: *Nervous Child* 2, pp. 217–250.

KNUTH, Donald E. (1997). *The Art of Computer Programming*. Addison-Wesley Professional. ISBN: 978-0-201-89683-1.

KOMIYA, Hidetoshi (1988). "Elementary proof for Sion's minimax theorem". In: *Kodai Mathematical Journal* 11.1, pp. 5–7. ISSN: 0386-5991. DOI: 10.2996/kmj/1138038812. URL: http://projecteuclid.org/getRecord?id=euclid.kmj/1138038812.

KOOPMAN, B. O. (1936). "On distributions admitting a sufficient statistic". In: *Transactions of the American Mathematical Society* 39.3, pp. 399–399. ISSN: 0002-9947. DOI: 10.1090/S0002-9947-1936-1501854-3. URL: http://www.ams.org/jourcgi/jour-getitem?pii=S0002-9947-1936-1501854-3.

KORBEL, Jan O., Alexander Eckehart URBAN, Fabian GRUBERT, Jiang DU, Thomas E. ROYCE, Peter STARR, Guoneng ZHONG, Beverly S. EMANUEL, Sherman M. WEISSMAN, Michael SNYDER & Mark B. GERSTEIN (2007). "Systematic prediction and validation of breakpoints associated with copy-number variants in the human genome." In: *Proceedings of the National Academy of Sciences of the United States of America* 104.24, pp. 10110–10115. ISSN: 0027-8424. DOI: 10.1073/pnas.0703834104. PMID: 17551006. URL: http://www.pnas.org/content/104/24/10110.abstract.

KUHN, H. W. (1955). "The Hungarian method for the assignment problem". In: *Naval Research Logistics Quarterly* 2.1-2, pp. 83–97. ISSN: 00281441. DOI: 10.1002/nav.3800020109. URL: http://doi.wiley.com/10.1002/nav.3800020109.

LAI, Weil R., Mark D. JOHNSON, Raju KUCHERLAPATI & Peter J. PARK (2005). "Comparative analysis of algorithms for identifying amplifications and deletions in array CGH data". In: *Bioinformatics* 21.19, pp. 3763–3770. DOI: 10.1093/bioinformatics/bti611. PMID: 16081473. URL: http://bioinformatics.oxfordjournals.org/content/21/19/3763.short.

Lawrence, Michael S., Petar Stojanov, Craig H. Mermel, James T. Robinson, Levi A. Garraway, Todd R. Golub, Matthew Meyerson, Stacey B. Gabriel, Eric S. Lander & Gad Getz (2014). "Discovery and saturation analysis of cancer genes across 21 tumour types". In: *Nature* 505.7484, pp. 495–501. doi: 10.1038/nature12912. url: http://www.nature.com/doifinder/10.1038/nature12912.

Leiserson, Mark D. M., Fabio Vandin, Hsin-Ta Wu, Jason R. Dobson, Jonathan V. Eldridge, Jacob L. Thomas, Alexandra Papoutsaki, Younhun Kim, Beifang Niu, Michael McLellan, Michael S. Lawrence, Abel Gonzalez-Perez, David Tamborero, Yuwei Cheng, Gregory A. Ryslik, Nuria Lopez-Bigas, Gad Getz, Li Ding & Benjamin J. Raphael (2014). "Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes". In: *Nature Genetics* 47.2, pp. 106–114. doi: 10.1038/ng.3168. url: http://www.nature.com/doifinder/10.1038/ng.3168.

Lewis, J. P. (1995). "Fast template matching". In: *Vision Interface 95*. Quebec City: Canadian Image Processing and Pattern Recognition Society, pp. 120–123. url: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.157.3888.

Li, H. & R. Durbin (2009). "Fast and accurate short read alignment with Burrows-Wheeler transform". In: *Bioinformatics* 25.14, pp. 1754–1760. issn: 1367-4803. doi: 10.1093/bioinformatics/btp324. url: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btp324.

Liu, Jun, Jaaved Mohammed, James Carter, Sanjay Ranka, Tamer Kahveci & Michael Baudis (2006). "Distance-based clustering of CGH data." In: *Bioinformatics (Oxford, England)* 22.16, pp. 1971–8. issn: 1367-4811. doi: 10.1093/bioinformatics/btl185. PMID: 16705014. url: http://bioinformatics.oxfordjournals.org/content/22/16/1971.short.

Lovász, László (1993). *Combinatorial problems and exercises*. American Mathematical Society, p. 639. isbn: 978-0-8218-4262-1.

Luo, J., M. Schumacher, A. Scherer, D. Sanoudou, D. Megherbi, T. Davison, T. Shi, W. Tong, L. Shi, H. Hong, C. Zhao, F. Elloumi, W. Shi, R. Thomas, S. Lin, G. Tillinghast, G. Liu, Y. Zhou, D. Herman, Y. Li, Y. Deng, H. Fang, P. Bushel, M. Woods & J. Zhang (2010). "A comparison of batch effect removal methods for enhancement of prediction performance using MAQC-II microarray gene expression data". In: *The Pharmacogenomics Journal* 10.4, pp. 278–291. issn: 1470-269X. doi: 10.1038/tpj.2010.57. PMID: 20676067. url: http://www.nature.com/articles/tpj201057.

Magi, A., L. Tattini, T. Pippucci, F. Torricelli & M. Benelli (2012). "Read count approach for DNA copy number variants detection". In: *Bioinformatics* 28.4, pp. 470–478. issn: 1367-4803. doi: 10.1093/bioinformatics/btr707. url: http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btr707.

Mahmud, Md Pavel & Alexander Schliep (2011). "Fast MCMC sampling for Hidden Markov Models to determine copy number variations". In: *BMC Bioinformatics* 12, p. 428. issn: 1471-2105. doi: 10.1186/1471-2105-12-428. PMID: 22047014. url: http://www.biomedcentral.com/1471-2105/12/428.

Malhotra, Dheeraj & Jonathan Sebat (2012). "CNVs: Harbingers of a Rare Variant Revolution in Psychiatric Genetics". In: *Cell* 148.6, pp. 1223–1241. doi: 10.1016/j.cell.2012.02.039.

Mallat, Stephane G. (1989). "Multiresolution approximations and wavelet orthonormal bases of $L^2(\mathbb{R})$". In: *Transactions of the American Mathematical Society* 315.1, pp. 69–69. issn: 0002-9947. doi: 10.1090/S0002-9947-1989-1008470-5. url: http://www.ams.org/tran/1989-315-01/S0002-9947-1989-1008470-5/.

– (2009). *A wavelet tour of signal processing: The Sparse way*. Burlington, MA: Academic Press. isbn: 978-0-12-374370-1. PMID: 15727384. url: http://dl.acm.org/citation.cfm?id=1525499.

Marioni, J. C., N. P. Thorne & S. Tavaré (2006). "BioHMM: a heterogeneous hidden Markov model for segmenting array CGH data." In: *Bioinformatics* 22.9, pp. 1144–6. issn: 1367-4803. doi: 10.1093/bioinformatics/btl089. PMID: 16533818. url: http://bioinformatics.oxfordjournals.org/content/22/9/1144.long.

Massart, Pascal (2003). "Concentration inequalities and model selection". In: *Lecture Notes in Mathematics* 1896, pp. 1–324. issn: 00758434. doi: 10.1007/978-3-540-48503-2. url: http://link.springer.com/10.1007/978-3-540-48503-2.

McKenna, Aaron, Matthew Hanna, Eric Banks, Andrey Sivachenko, Kristian Cibulskis, Andrew Kernytsky, Kiran Garimella, David Altshuler, Stacey Gabriel, Mark Daly & Mark A. DePristo (2010). "The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data." In: *Genome research* 20.9, pp. 1297–303. issn: 1549-5469. doi: 10.1101/gr.107524.110. PMID: 20644199. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2928508.

McLare, William, Laurent Gil, Sarah E. Hunt, Harpreet Singh Riat, Graham R. S. Ritchie, Anja Thormann, Paul Flicek & Fiona Cunningham (2016). "The Ensembl Variant Effect Predictor". In: *Genome Biology* 17.22. issn: 0028-0836. doi: 10.1038/513S8a. url: http://www.nature.com/doifinder/10.1038/513S8a.

Merikangas, Alison K., Aiden P. Corvin & Louise Gallagher (2009). "Copy-number variants in neurodevelopmental disorders: promises and challenges." In: *Trends in Genetics* 25.12, pp. 536–44. issn: 0168-9525. doi: 10.1016/j.tig.2009.10.006. PMID: 19910074. url: http://www.sciencedirect.com/science/article/pii/S016895250900211X.

Metzker, Michael L. (2010). "Sequencing technologies — the next generation". In: *Nature Reviews Genetics* 11.1, pp. 31–46. issn: 1471-0056. doi: 10.1038/nrg2626. url: http://www.nature.com/doifinder/10.1038/nrg2626.

Meyer, Yves & David H. Salinger (1992). *Wavelets and operators*. Cambridge University Press, p. 223. isbn: 978-0-521-45869-6.

Moya, Pablo R., Nicholas H. Dodman, Kiara R. Timpano, Liza M. Rubenstein, Zaker Rana, Ruby L. Fried, Louis F. Reichardt, Gary A. Heiman, Jay A. Tischfield, Robert A. King, Marzena Galdzicka, Edward I. Ginns & Jens R. Wendland (2013). "Rare missense neuronal cadherin gene (CDH2) variants in specific obsessive-compulsive disorder and Tourette disorder phenotypes". In: *European Journal of Human Genetics* 21.8, pp. 850–854. issn: 1018-4813. doi: 10.1038/ejhg.2012.245. url: http://www.nature.com/doifinder/10.1038/ejhg.2012.245.

Murphy, Kevin P. (2012). *Machine Learning. A Probabilistic Perspective*. Massachusetts Institute of Technology. isbn: 978-0-262-01802-9.

Nag, Abhishek, Elena G. Bochukova, Barbara Kremeyer, Desmond D. Campbell, Heike Muller, Ana V. Valencia-Duarte, Julio Cardona, Isabel C. Rivas, Sandra C. Mesa, Mauricio Cuartas, Jharley Garcia, Gabriel Bedoya, William Cornejo, Luis D. Herrera, Roxana Romero, Eduardo Fournier, Victor I. Reus, Thomas L. Lowe, I. Sadaf Farooqi, Carol A. Mathews, Lauren M. McGrath, Dongmei Yu, Ed Cook, Kai Wang, Jeremiah M. Scharf, David L. Pauls, Nelson B. Freimer, Vincent Plagnol & Andrés Ruiz-Linares (2013). "CNV Analysis in Tourette Syndrome Implicates Large Genomic Rearrangements in COL8A1 and NRXN1". In: *PLoS ONE* 8.3. Ed. by Ge Zhang, e59061. issn: 1932-6203. doi: 10.1371/journal.pone.0059061. url: http://dx.plos.org/10.1371/journal.pone.0059061.

Nakagawa, H., C. P. Wardell, M. Furuta, H. Taniguchi & A. Fujimoto (2015). "Cancer whole-genome sequencing: present and future". In: *Oncogene* 34.49, pp. 5943–5950. doi: 10.1038/onc.2015.90. PMID: 25823020. url: http://www.nature.com/doifinder/10.1038/onc.2015.90.

Nannya, Yasuhito, Masashi Sanada, Kumi Nakazaki, Noriko Hosoya, Lili Wang, Akira Hangaishi, Mineo Kurokawa, Shigeru Chiba, Dione K. Bailey, Giulia C. Kennedy & Seishi Ogawa (2005). "A robust algorithm for copy number detection using high-density oligonucleotide single nucleotide polymorphism genotyping arrays." In: *Cancer Research* 65.14, pp. 6071–9. issn: 0008-5472. doi: 10.1158/0008-5472.CAN-05-0465. PMID: 16024607. url: http://cancerres.aacrjournals.org/content/65/14/6071.short.

Navin, Nicholas E. (2015). "The first five years of single-cell cancer genomics and beyond." In: *Genome research* 25.10, pp. 1499–507. doi: 10.1101/gr.191098.115. PMID: 26430160. url: http://www.ncbi.nlm.nih.gov/pubmed/26430160.

Neyman, J. (1936). "Su un teorema concernente le cosiddette statistiche sufficienti". In: *Giornale dell'Istituto Italiano Italiano degli Attuari* 6, pp. 320–334.

Nguyen, Nha, Heng Huang, Soontorn Oraintara & An Vo (2007). "A New Smoothing Model for Analyzing Array CGH Data". In: *Proceedings of the 7th IEEE International Conference on Bioinformatics and Bioengineering*. Boston, MA. doi: 10.1109/BIBE.2007.4375683. url: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4375683.

– (2010). "Stationary wavelet packet transform and dependent laplacian bivariate shrinkage estimator for array-CGH data smoothing." en. In: *Journal of Computational Biology* 17.2, pp. 139–52. issn: 1557-8666. doi: 10.1089/cmb.2009.0013. PMID: 20078226. url: http://online.liebertpub.com/doi/abs/10.1089/cmb.2009.0013.

O'Rourke, Julia A., Jeremiah M. Scharf, Dongmei Yu & David L. Pauls (2009). "The genetics of Tourette syndrome: a review." In: *Journal of psychosomatic research* 67.6, pp. 533–45. issn: 1879-1360. doi: 10.1016/j.jpsychores.2009.06.006. PMID: 19913658. url: https://www.sciencedirect.com/science/article/pii/S002239990900258X.

Olshen, Adam B. & E. S. Venkatraman (2002). "Change-point analysis of array-based comparative genomic hybridization data". In: *ASA Proceedings of the Joint Statistical Meetings*, pp. 2530–2535.

Olshen, Adam B., E. S. Venkatraman, Robert Lucito & Michael Wigler (2004). "Circular binary segmentation for the analysis of array-based DNA copy number data." In: *Biostatistics (Oxford, England)* 5.4, pp. 557–572. issn: 1465-4644. doi: 10.1093/biostatistics/kxh008. PMID: 15475419. url: http://biostatistics.oxfordjournals.org/content/5/4/557.short.

Özgür, Arzucan, Levent Özgür & Tunga Güngör (2005). "Text Categorization with Class-Based and Corpus-Based Keyword Selection". In: *Proceedings of the 20th International Conference on Computer and Information Sciences* 3733, pp. 606–615. issn: 03029743. doi: 10.1007/11569596. url: http://link.springer.com/chapter/10.1007/11569596_63.

Pabinger, Stephan, Andreas Dander, Maria Fischer, Rene Snajder, Michael Sperk, Mirjana Efremova, Birgit Krabichler, Michael R. Speicher, Johannes Zschocke & Zlatko Trajanoski (2014). "A survey of tools for variant analysis of next-generation genome sequencing data". In: *Briefings in Bioinformatics* 15.2, pp. 256–278. issn: 14774054. doi: 10.1093/bib/bbs086. arXiv: 209. url: http://bib.oxfordjournals.org/content/15/2/256.full.

Pauls, D. L., K. E. Towbin, J. F. Leckman, G. E. Zahner & D. J. Cohen (1986). "Gilles de la Tourette's syndrome and obsessive-compulsive disorder. Evidence supporting a genetic relationship." In: *Archives of general psychiatry* 43.12, pp. 1180–2. issn: 0003-990X. PMID: 3465280. url: http://www.ncbi.nlm.nih.gov/pubmed/3465280.

Picard, Franck, Stephane Robin, Marc Lavielle, Christian Vaisse & Jean-Jacques Daudin (2005). "A statistical approach for array CGH data analysis." In: *BMC Bioinformatics* 6.1, p. 27. issn: 1471-2105. doi: 10.1186/1471-2105-6-27. PMID: 15705208. url: http://www.biomedcentral.com/1471-2105/6/27.

Pinto, Dalila, Katayoon Darvishi, Xinghua Shi, Diana Rajan, Diane Rigler, Tom Fitzgerald, Anath C. Lionel, Bhooma Thiruvahindrapuram, Jeffrey R. Macdonald, Ryan Mills, Aparna Prasad, Kristin Noonan, Susan Gribble, Elena Prigmore, Patricia K. Donahoe, Richard S. Smith, Ji Hyeon Park, Matthew E. Hurles, Nigel P. Carter, Charles Lee, Stephen W. Scherer & Lars Feuk (2011). "Comprehensive assessment of array-based platforms and calling algorithms for detection of copy number variants." In: *Nature Biotechnology* 29.6, pp. 512–520. issn: 1546-1696. doi: 10.1038/nbt.1852. PMID: 21552272. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3270583.

Pirooznia, Mehdi, Fernando S. Goes & Peter P. Zandi (2015). "Whole-genome CNV analysis: advances in computational approaches". In: *Frontiers in Genetics* 06.MAR, p. 138. issn: 1664-8021. doi: 10.3389/fgene.2015.00138. PMID: 25918519. url: http://journal.frontiersin.org/article/10.3389/fgene.2015.00138/abstract.

Pitman, E. J. G., J. Wishart & R. A. Fisher (1936). "Sufficient statistics and intrinsic accuracy". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 32.04, p. 567. issn: 0305-0041. doi: 10.1017/S0305004100019307. url: http://www.journals.cambridge.org/abstract_S0305004100019307.

Rabiner, L. R. & B. H. Juang (1986). "An introduction to hidden Markov models". In: *IEEE ASSP Magazine* 3.1, pp. 4–16. issn: 0740-7467. doi: 10.1109/MASSP.1986.1165342. url: http://ieeexplore.ieee.org/document/1165342/.

Rabiner, Lawrence R. (1989). "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition". In: *Proceedings of the IEEE* 77.2, pp. 257–286. issn: 15582256. doi: 10.1109/5.18626. arXiv: 1011.1669v3. url: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=18626.

Redon, Richard, Shumpei Ishikawa, Karen R. Fitch, Lars Feuk, George H. Perry, T. Daniel Andrews, Heike Fiegler, Michael H. Shapero, Andrew R. Carson, Wenwei Chen, Eun Kyung Cho, Stephanie Dallaire, Jennifer L. Freeman, Juan R. González, Mònica Gratacòs, Jing Huang, Dimitrios Kalaitzopoulos, Daisuke Komura, Jeffrey R. MacDonald, Christian R. Marshall, Rui Mei, Lyndal Montgomery, Kunihiro Nishimura, Kohji Okamura, Fan Shen, Martin J. Somerville, Joelle Tchinda, Armand Valsesia, Cara Woodwark, Fengtang Yang, Junjun Zhang, Tatiana Zerjal, Jane Zhang, Lluis Armengol, Donald F. Conrad, Xavier Estivill, Chris Tyler-Smith, Nigel P. Carter, Hiroyuki Aburatani, Charles Lee, Keith W. Jones, Stephen W. Scherer & Matthew E. Hurles (2006). "Global variation in copy number in the human genome". In: *Nature* 444.7118, pp. 444–454. issn: 0028-0836. doi: 10.1038/nature05329. PMID: 17122850. url: http://www.nature.com/doifinder/10.1038/nature05329.

Renaud, Gabriel, Martin Kircher, Udo Stenzel & Janet Kelso (2013). "freeIbis: an efficient basecaller with calibrated quality scores for Illumina sequencers". In: *Bioinformatics* 29.9, pp. 1208–1209. issn: 1460-2059. doi: 10.1093/bioinformatics/btt117. url: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btt117.

Renaud, Gabriel, Udo Stenzel & Janet Kelso (2014). "leeHom: adaptor trimming and merging for Illumina sequencing reads". In: *Nucleic Acids Research* 42.18, e141–e141. issn: 1362-4962. doi: 10.1093/nar/gku699. url: https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gku699.

Renaud, Gabriel, Udo Stenzel, Tomislav Maricic, Victor Wiebe & Janet Kelso (2015). "deML: robust demultiplexing of Illumina sequences using a likelihood-based approach". In: *Bioinformatics* 31.5, pp. 770–772. issn: 1460-2059. doi: 10.1093/bioinformatics/btu719. url: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu719.

Richards, A. Brent, Tracy A. Scheel, Kan Wang, Mark Henkemeyer & Lawrence F. Kromer (2007). "EphB1 null mice exhibit neuronal loss in substantia nigra pars reticulata and spontaneous locomotor hyperactivity". In: *European Journal of Neuroscience* 25.9, pp. 2619–2628. issn: 0953816X. doi:

10.1111/j.1460-9568.2007.05523.x. PMID: 17561836. URL: http://doi.wiley.com/10.1111/j.1460-9568.2007.05523.x.

Robert, Christian P. (2007). *The Bayesian Choice*. Second edition. Springer. ISBN: 978-0-387-71598-8. URL: https://www.springer.com/gp/book/9780387952314.

Rueda, Oscar M. & Ramon Diaz-Uriarte (2009). "RJaCGH: Bayesian analysis of aCGH arrays for detecting copy number changes and recurrent regions." In: *Bioinformatics* 25.15, pp. 1959–1960. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btp307. PMID: 19420051. URL: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2712338.

Sánchez-Villagra, Marcelo R., Madeleine Geiger & Richard A. Schneider (2016). "The taming of the neural crest: a developmental perspective on the origins of morphological covariation in domesticated mammals". In: *Royal Society Open Science* 3.6.

Schoumans, J., C. Ruivenkamp, E. Holmberg, M. Kyllerman, B.-M. Anderlid & M. Nordenskjöld (2005). "Detection of chromosomal imbalances in children with idiopathic mental retardation by array based comparative genomic hybridisation (array-CGH)". In: *Journal of Medical Genetics* 42.9, pp. 699–705. ISSN: 1468-6244. DOI: 10.1136/jmg.2004.029637. PMID: 16141005. URL: http://jmg.bmj.com/content/42/9/699.long.

Scott, Steven L. (2002). "Bayesian methods for hidden Markov models: Recursive computing in the 21st century". en. In: *Journal of the American Statistical Association* 97.457, pp. 337–351. ISSN: 01621459. DOI: 10.1198/016214502753479464. URL: http://www.jstor.org/stable/3085787.

Sebat, Jonathan, B. Lakshmi, Dheeraj Malhotra, Jennifer Troge, Christa Lese-Martin, Tom Walsh, Boris Yamrom, Seungtai Yoon, Alex Krasnitz, Jude Kendall, Anthony Leotta, Deepa Pai, Ray Zhang, Yoon-Ha Lee, James Hicks, Sarah J. Spence, Annette T. Lee, Kaija Puura, Terho Lehtimäki, David Ledbetter, Peter K. Gregersen, Joel Bregman, James S. Sutcliffe, Vaidehi Jobanputra, Wendy Chung, Dorothy Warburton, Mary-Claire King, David Skuse, Daniel H. Geschwind, T. Conrad Gilliam, Kenny Ye & Michael Wigler (2007). "Strong Association of De Novo Copy Number Mutations with Autism". In: *Science* 316.5823, pp. 445–449. ISSN: 0036-8075. DOI: 10.1126/science.1138659. PMID: 17363630. URL: http://www.sciencemag.org/content/316/5823/445.abstract.

Sebat, Jonathan, B. Lakshmi, Jennifer Troge, Joan Alexander, Janet Young, Pär Lundin, Susanne Månér, Hillary Massa, Megan Walker, Maoyen Chi, Nicholas Navin, Robert Lucito, John Healy, James Hicks, Kenny Ye, Andrew Reiner, T. Conrad Gilliam, Barbara Trask, Nick Patterson, Anders Zetterberg & Michael Wigler (2004). "Large-Scale Copy Number Polymorphism in the Human Genome". In: *Science* 305.5683, pp. 525–528. ISSN: 0036-8075. DOI: 10.1126/science.1098918. PMID: 15273396. URL: http://www.sciencemag.org/cgi/doi/10.1126/science.1098918.

Sener, Elif Funda (2014). "Association of Copy Number Variations in Autism Spectrum Disorders: A Systematic Review". In: *Chinese Journal of Biology* 2014, pp. 1–9. ISSN: 2314-7474. DOI: 10.1155/2014/713109. URL: http://www.hindawi.com/journals/cjb/2014/713109/.

Shah, Sohrab P., Wan L. Lam, Raymond T. Ng & Kevin P. Murphy (2007). "Modeling recurrent DNA copy number alterations in array CGH data." In: *Bioinformatics* 23.13, pp. i450–8. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btm221. PMID: 17646330. URL: http://bioinformatics.oxfordjournals.org/content/23/13/i450.short.

Shah, Sohrab P., Xiang Xuan, Ron J. DeLeeuw, Mehrnoush Khojasteh, Wan L. Lam, Raymond Ng & Kevin P. Murphy (2006). "Integrating copy number polymorphisms into array CGH analysis using a robust HMM." In: *Bioinformatics* 22.14, e431–9. ISSN: 1367-4811. DOI: 10.1093/bioinformatics/btl238. PMID: 16873504. URL: http://bioinformatics.oxfordjournals.org/content/22/14/e431.

Sharp, Andrew J., Devin P. Locke, Sean D. McGrath, Ze Cheng, Jeffrey A. Bailey, Rhea U. Vallente, Lisa M. Pertz, Royden A. Clark, Stuart Schwartz, Rick Segraves, Vanessa V. Oseroff, Donna G. Albertson, Daniel Pinkel & Evan E. Eichler (2005). "Segmental Duplications and Copy-Number Variation in the Human Genome". In: *The American Journal of Human Genetics* 77.1, pp. 78–88. issn: 00029297. doi: 10.1086/431652. PMID: 15918152. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1226196.

Shlien, Adam & David Malkin (2009). "Copy number variations and cancer." In: *Genome medicine* 1.6, p. 62. doi: 10.1186/gm62. PMID: 19566914. url: http://www.ncbi.nlm.nih.gov/pubmed/19566914.

Sion, Maurice (1958). *On general minimax theorems*. Vol. 8. 1. Pacific Journal of Mathematics, pp. 171–176. url: https://msp.org/pjm/1958/8-1/p14.xhtml.

Snijders, Antoine M., Jane Fridlyand, Dorus A. Mans, Richard Segraves, Ajay N. Jain, Daniel Pinkel & Donna G. Albertson (2003). "Shaping of tumor and drug-resistant genomes by instability and selection." In: *Oncogene* 22.28, pp. 4370–9. issn: 0950-9232. doi: 10.1038/sj.onc.1206482. PMID: 12853973. url: http://www.ncbi.nlm.nih.gov/pubmed/12853973.

Snijders, Antoine M., Norma Nowak, Richard Segraves, Stephanie Blackwood, Nils Brown, Jeffrey Conroy, Greg Hamilton, Anna Katherine Hindle, Bing Huey, Karen Kimura, Sindy Law, Ken Myambo, Joel Palmer, Bauke Ylstra, Jingzhu Pearl Yue, Joe W. Gray, Ajay N. Jain, Daniel Pinkel & Donna G. Albertson (2001). "Assembly of microarrays for genome-wide measurement of DNA copy number." In: *Nature Genetics* 29.3, pp. 263–264. issn: 1061-4036. doi: 10.1038/ng754. PMID: 11687795. url: https://www.nature.com/articles/ng754.

Strang, Gilbert. & T. Nguyen (1997). *Wavelets and filter banks*. Wellesley-Cambridge Press, p. 520. isbn: 978-0-9614088-7-9.

Sturtevant, Alfred Henry (1929). "The claret mutant type of Drosophila simulans: a study of chromosome elimination and cell-lineage". In: *Zeitschrift für wissenschaftliche Zoologie* 135, pp. 323–356.

Sweldens, Wim (1995). "Lifting scheme: a new philosophy in biorthogonal wavelet constructions". In: ed. by Andrew F. Laine & Michael A. Unser. International Society for Optics and Photonics, pp. 68–79. doi: 10.1117/12.217619. url: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=1007578.

– (1998). "The Lifting Scheme: A Construction of Second Generation Wavelets". en. In: *SIAM Journal on Mathematical Analysis* 29.2, pp. 511–546. issn: 0036-1410. doi: 10.1137/S0036141095289051. url: http://epubs.siam.org/doi/abs/10.1137/S0036141095289051.

Tetreault, Martine, Eric Bareke, Javad Nadaf, Najmeh Alirezaie & Jacek Majewski (2015). "Whole-exome sequencing as a diagnostic tool: current challenges and future opportunities". In: *Expert Review of Molecular Diagnostics*, pp. 1–12. issn: 1473-7159. doi: 10.1586/14737159.2015.1039516. url: http://informahealthcare.com/doi/abs/10.1586/14737159.2015.1039516.

The International HapMap Consortium (2005). "A haplotype map of the human genome". In: *Nature* 437.7063, pp. 1299–1320. issn: 0028-0836. doi: 10.1038/nature04226. url: http://www.nature.com/doifinder/10.1038/nature04226.

The International SNP Map Working Group, Ravi Sachidanandam, David Weissman, Steven C. Schmidt, Jerzy M. Kakol, Lincoln D. Stein, Gabor Marth, Steve Sherry, James C. Mullikin, Beverley J. Mortimore, David L. Willey, Sarah E. Hunt, Charlotte G. Cole, Penny C. Coggill, Catherine M. Rice, Zemin Ning, Jane Rogers, David R. Bentley, Pui-Yan Kwok, Elaine R. Mardis, Raymond T. Yeh, Brian Schultz, Lisa Cook, Ruth Davenport, Michael Dante, Lucinda Fulton, LaDeana Hillier, Robert H. Waterston, John D. McPherson, Brian Gilman, Stephen Schaffner, William J. Van Etten, David Reich, John Higgins, Mark J. Daly, Brendan Blumenstiel, Jennifer Baldwin, Nicole Stange-

Thomann, Michael C. Zody, Lauren Linton, Eric S. Lander & David Altshuler (2001). "A map of human genome sequence variation containing 1.42 million single nucleotide polymorphisms". In: *Nature* 409.6822, pp. 928–933. issn: 0028-0836. doi: 10.1038/35057149. url: http://www.nature.com/doifinder/10.1038/35057149.

Trut, L. N., I. Z. Plyusnina & I. N. Oskina (2004). "An Experiment on Fox Domestication and Debatable Issues of Evolution of the Dog". In: *Russian Journal of Genetics* 40.6, pp. 644–655. issn: 1022-7954. doi: 10.1023/B:RUGE.0000033312.92773.c1. url: http://link.springer.com/10.1023/B:RUGE.0000033312.92773.c1.

Trut, Lyudmila, Irina Oskina & Anastasiya Kharlamova (2009). "Animal evolution during domestication: the domesticated fox as a model." In: *BioEssays : news and reviews in molecular, cellular and developmental biology* 31.3, pp. 349–60. issn: 1521-1878. doi: 10.1002/bies.200800070. PMID: 19260016. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2763232.

Tsourakakis, Charalampos E., Richard Peng, Maria A. Tsiarli, Gary L. Miller & Russell Schwartz (2011). "Approximation algorithms for speeding up dynamic programming and denoising aCGH data". In: *Journal of Experimental Algorithmics* 16, p. 1.1. issn: 10846654. doi: 10.1145/1963190.2063517. url: http://dl.acm.org/citation.cfm?id=1963190.2063517.

Tuzun, Eray, Andrew J. Sharp, Jeffrey A. Bailey, Rajinder Kaul, V. Anne Morrison, Lisa M. Pertz, Eric Haugen, Hillary Hayden, Donna Albertson, Daniel Pinkel, Maynard V. Olson & Evan E. Eichler (2005). "Fine-scale structural variation of the human genome". In: *Nature Genetics* 37.7, pp. 727–732. issn: 1061-4036. doi: 10.1038/ng1562. url: http://www.nature.com/doifinder/10.1038/ng1562.

Tyson, C., C. Harvard, R. Locker, J. M. Friedman, S. Langlois, M. E. S. Lewis, M. Van Allen, M. Somerville, L. Arbour, L. Clarke, B. McGilivray, S. L. Yong, J. Siegel-Bartel & E. Rajcan-Separovic (2005). "Submicroscopic deletions and duplications in individuals with intellectual disability detected by array-CGH". In: *American Journal of Medical Genetics Part A* 139A.3, pp. 173–185. issn: 1552-4825. doi: 10.1002/ajmg.a.31015. PMID: 16283669. url: http://doi.wiley.com/10.1002/ajmg.a.31015.

Valsesia, Armand, Aurélien Macé, Sébastien Jacquemont, Jacques S. Beckmann & Zoltán Kutalik (2013). "The Growing Importance of CNVs: New Insights for Detection and Clinical Interpretation." English. In: *Frontiers in Genetics* 4.May, p. 92. issn: 1664-8021. doi: 10.3389/fgene.2013.00092. PMID: 23750167. url: http://journal.frontiersin.org/article/10.3389/fgene.2013.00092/full.

Van de Wiel, Mark A. & Wessel N. van Wieringen (2007). "CGHregions: Dimension reduction for array CGH data with minimal information loss". In: *Cancer Informatics* 3, pp. 55–63. issn: 11769351. PMID: 19455235. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2675846.

Van der Auwera, Geraldine A., Mauricio O. Carneiro, Christopher Hartl, Ryan Poplin, Guillermo del Angel, Ami Levy-Moonshine, Tadeusz Jordan, Khalid Shakir, David Roazen, Joel Thibault, Eric Banks, Kiran V. Garimella, David Altshuler, Stacey Gabriel & Mark A. DePristo (2013). "From FastQ Data to High-Confidence Variant Calls: The Genome Analysis Toolkit Best Practices Pipeline". In: *Current Protocols in Bioinformatics*. Hoboken, NJ, USA: John Wiley & Sons, Inc., pp. 11.10.1–11.10.33. isbn: 978-0-471-25095-1. doi: 10.1002/0471250953.bi1110s43. url: http://doi.wiley.com/10.1002/0471250953.bi1110s43.

Van Wieringen, Wessel N., Mark A. van ve Wiel & Bauke Ylstra (2008). "Weighted clustering of called array CGH data". en. In: *Biostatistics* 9.3, pp. 484–500. issn: 14654644. doi: 10.1093/biostatistics/kxm048. arXiv: 1109.1844v1. url: http://biostatistics.oxfordjournals.org/content/9/3/484.full.

Venkatraman, E. S. & Adam B. Olshen (2007). "A faster circular binary segmentation algorithm for the analysis of array CGH data." In: *Bioinformatics* 23.6, pp. 657–63. issn: 1367-4811. doi:

10.1093/bioinformatics/btl646. PMID: 17234643. URL: http://bioinformatics.oxfordjournals.org/content/23/6/657.short.

VITERBI, A. (1967). "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm". In: *IEEE Transactions on Information Theory* 13.2, pp. 260–269. ISSN: 0018-9448. DOI: 10.1109/TIT.1967.1054010. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1054010.

VOGELSTEIN, B., Nickolas PAPADOPOULOS, Victor E. VELCULESCU, S. ZHOU, L. A. DIAZ & K. W. KINZLER (2013). "Cancer Genome Landscapes". In: *Science* 339.6127, pp. 1546–1558. ISSN: 0036-8075. DOI: 10.1126/science.1235122. PMID: 23539594. URL: http://www.sciencemag.org/cgi/doi/10.1126/science.1235122.

WALD, Abraham (1945). "Statistical Decision Functions Which Minimize the Maximum Risk". In: *The Annals of Mathematics* 46.2, p. 265. ISSN: 0003486X. DOI: 10.2307/1969022. URL: http://www.jstor.org/stable/1969022.

WANG, Jing, Dexter DUNCAN, Zhiao SHI & Bing ZHANG (2013). "WEB-based GEne SeT AnaLysis Toolkit (WebGestalt): update 2013". In: *Nucleic Acids Research* 41.W1, W77–W83. ISSN: 1362-4962. DOI: 10.1093/nar/gkt439. PMID: 23703215. URL: https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkt439.

WANG, Junbai, Leonardo A. MEZA-ZEPEDA, Stine H. KRESSE & Ola MYKLEBOST (2004). "M-CGH: analysing microarray-based CGH experiments." In: *BMC Bioinformatics* 5.1, p. 74. ISSN: 1471-2105. DOI: 10.1186/1471-2105-5-74. PMID: 15189572. URL: http://www.biomedcentral.com/1471-2105/5/74.

WANG, Yuhang & Siling WANG (2007). "A novel stationary wavelet denoising algorithm for array-based DNA Copy Number data." In: *International Journal of Bioinformatics Research and Applications* 3.x, pp. 206–222. ISSN: 1744-5485. DOI: 10.1504/IJBRA.2007.013603. PMID: 18048189.

WIEDENHOEFT, John, Eric BRUGEL & Alexander SCHLIEP (2016a). "Fast Bayesian Inference of Copy Number Variants using Hidden Markov Models with Wavelet Compression". In: *PLOS Computational Biology* 5 (12). DOI: 10.1371/journal.pcbi.1004871. URL: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1004871.

– (2016b). "Fast Bayesian Inference of Copy Number Variants using Hidden Markov Models with Wavelet Compression". In: *Research in Computational Molecular Biology: 20th Annual Conference, RECOMB 2017*. This talk was a parallel submission to the PLOS paper of the same title. ISBN: 978-3-319-31957-5.

– (2016c). "Fast Bayesian Inference of Copy Number Variants using Hidden Markov Models with Wavelet Compression". In: *PLOS Computational Biology* 12.5. Ed. by Paul P. GARDNER, e1004871. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1004871. URL: http://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1004871.

WIEDENHOEFT, John, Alex CAGAN, Rimma KOZHEMJAKINA, Rimma GULEVICH & Alexander SCHLIEP (2017). "Locating CNV candidates in WGS data using wavelet-compressed Bayesian HMM". In: *ISMB/ECCB 2017*. Conference poster. URL: https://f1000research.com/posters/6-1418.

WIEDENHOEFT, John, Alex CAGAN, Rimma KOZHEMYAKINA, Rimma GULEVICH & Alexander SCHLIEP (2018). "Bayesian localization of CNV candidates in WGS data within minutes". In: *Submitted to BMC Algorithms for Molecular Biology*.

WIEDENHOEFT, John & Alexander SCHLIEP (2017). "Using HaMMLET for Bayesian Segmentation of WGS Read-Depth Data". In: *Copy Number Variants: Methods and Protocols*. Ed. by Derek BICKHART. Methods in Molecular Biology 1833. Springer, pp. 83–93. DOI: 10.1007/978-1-4939-8666-8_6. URL: https://link.springer.com/protocol/10.1007%2F978-1-4939-8666-8_6.

Wiedenhoeft, John & Alexander Schliep (2018). "Decision-theoretic foundation of Bayesian HMM inference under Haar wavelet compression". In: *Submitted to NIPS 2018*.

Wilkins, Adam S., Richard W. Wrangham & W. Tecumseh Fitch (2014). "The "Domestication Syndrome" in Mammals: A Unified Explanation Based on Neural Crest Cell Behavior and Genetics". In: *Genetics* 197.3.

Willenbrock, Hanni & Jane Fridlyand (2005). "A comparison study: applying segmentation to array CGH data for downstream analyses." In: *Bioinformatics* 21.22, pp. 4084–91. issn: 1367-4803. doi: 10.1093/bioinformatics/bti677. PMID: 16159913. url: http://bioinformatics.oxfordjournals.org/content/21/22/4084.short.

Wineinger, Nathan E., Richard E. Kennedy, Stephen W. Erickson, Mary K. Wojczynski, Carl E. Bruder & Hemant K. Tiwari (2008). "Statistical issues in the analysis of DNA Copy Number Variations." In: *International journal of computational biology and drug design* 1.4, pp. 368–95. issn: 1756-0756. doi: 10.1504/IJCBDD.2008.022208. PMID: 19774103. url: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2747762.

Xu, Jie, Lonnie Zwaigenbaum, Peter Szatmari & Stephen Scherer (2004). "Molecular Cytogenetics of Autism". In: *Current Genomics* 5.4, pp. 347–364. issn: 13892029. doi: 10.2174/1389202043349246. url: http://www.eurekaselect.com/openurl/content.php?genre=article&issn=1389-2029&volume=5&issue=4&spage=347.

Yin, Xiao-lin & Jing Li (2009). "A general graphical framework for detecting copy number variations". In: *8th Annual International Conference on Computational Systems Bioinformatics*. Life Sciences Society. url: http://www.csb2009a.org/pdf/060Li.pdf.

Zhang, Bing, Stefan Kirov & Jay Snoddy (2005). "WebGestalt: an integrated system for exploring gene sets in various biological contexts". In: *Nucleic Acids Research* 33.Web Server, W741–W748. issn: 0305-1048. doi: 10.1093/nar/gki475. PMID: 15980575. url: https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gki475.

Zhao, Min, Qingguo Wang, Quan Wang, Peilin Jia & Zhongming Zhao (2013). "Computational tools for copy number variation (CNV) detection using next-generation sequencing data: features and perspectives - Springer". In: *BMC Bioinformatics* 14 Suppl 1.Suppl 11, S1. issn: 1471-2105. doi: 10.1186/1471-2105-14-S11-S1. PMID: 24564169. url: http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-14-S11-S1.

Zhao, X. (2004). "An Integrated View of Copy Number and Allelic Alterations in the Cancer Genome Using Single Nucleotide Polymorphism Arrays". In: *Cancer Research* 64.9, pp. 3060–3071. issn: 0008-5472. doi: 10.1158/0008-5472.CAN-03-3308. url: http://cancerres.aacrjournals.org/content/64/9/3060.short.